AIAA JOURNAL
Vol. 42, No. 8, August 2004

New Hybrid Runge–Kutta Methods for Unsteady Reactive Flow Simulation

Jack J. Yoh*
Lawrence Livermore National Laboratory, Livermore, California 94551
and
Xiaolin Zhong†
University of California, Los Angeles, Los Angeles, California 90095

In the numerical simulation of transient reacting flow, standard explicit calculation is prohibitively expensive because of the small time steps needed to address the stiffness of a governing differential system. To circumvent this, new hybrid implicit–explicit methods proposed treat the stiffness, whereas the underlying time-step control is governed by the Courant stability criterion. Because the coefficients of both explicit and implicit operations are entirely determined by solving the necessary conditions of accuracy and L stability without any extra assumptions, the methods are more generalized than other similar methods in the literature. Two families of semi-implicit Runge–Kutta schemes are developed for split differential equations in the form of $u' = f(t, u) + g(t, u)$, where f is treated explicitly and g is simultaneously treated implicitly. Like the rest of all the developed schemes, a low-storage family of semi-implicit schemes is also derived to be globally high-order accurate and L stable for implicit calculations. In a companion paper (Yoh, J. J., and Zhong, X., "New Hybrid Runge–Kutta Methods for Unsteady Reactive Flow Simulation: Applications," ***AIAA Journal*****, Vol. 42, No. 8, 2004, pp. 1601–1611) the new schemes are tested to solve a wide range of applications in high-speed flow physics involving combustion.**

Nomenclature

a_i, b_i	=	coefficients of low-storage semi-implicit Runge–Kutta (LSSIRK) schemes
b_{ij}	=	coefficients of explicit part of semi-implicit Runge–Kutta (SIRK) schemes
c_i	=	coefficients of implicit part of LSSIRK schemes, d_i
c_{ij}	=	coefficients of implicit part of SIRK schemes
$\bar{c}_i$	=	coefficients of implicit part of LSSIRK schemes
d_i	=	diagonal terms of c_{ij}
$\boldsymbol{F}$	=	flux vector
$f(x, t)$	=	nonstiff part of $\mathrm{d}u/\mathrm{d}t$
$\boldsymbol{f}(\boldsymbol{x}, t)$	=	nonstiff part of an ordinary differential equation (ODE) system $\mathrm{d}\boldsymbol{u}/\mathrm{d}t$
$g(u, t)$	=	stiff part of $\mathrm{d}u/\mathrm{d}t$
$\boldsymbol{g}(\boldsymbol{u}, t)$	=	stiff part of an ODE system $\mathrm{d}\boldsymbol{u}/\mathrm{d}t$
h	=	time step size
$\boldsymbol{J}$	=	Jacobian matrix of $\boldsymbol{g}$
k_i	=	intermediate functions for r-stage Runge–Kutta scheme
r	=	maximum stage index for r-stage Runge–Kutta schemes
r_i	=	time coefficients of explicit Runge–Kutta schemes
s_i	=	time coefficients of implicit Runge–Kutta schemes
t	=	independent or nonautonomous variable
$\boldsymbol{U}$	=	conservation variable vector
$u(t)$	=	dependent variable, scalar
$\boldsymbol{u}(t)$	=	dependent variable, vector
$\boldsymbol{W}$	=	source vector
x	=	spatial coordinate
α	=	wedge angle bounded by the axes of the stability plane
λ_f	=	eigenvalues of $\partial\boldsymbol{f}/\partial\boldsymbol{u}$
λ_g	=	eigenvalues of $\partial\boldsymbol{g}/\partial\boldsymbol{u}$
ω_i	=	weight coefficients of SIRK schemes

Subscripts

f	=	nonstiff term
g	=	stiff term
i, j	=	dummy indices

Introduction

THIS paper is concerned with robust and high-order-accurate methods for computing stiff systems of ordinary differential equations that can be additively separated into relatively stiff and nonstiff terms in the following form:

$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{u}, t) + \boldsymbol{g}(\boldsymbol{u}, t) \tag{1}$$

where $\boldsymbol{f}$ and $\boldsymbol{g}$ are vectors representing nonstiff and stiff terms, respectively, and not vice versa. The maximum magnitude of the eigenvalues of Jacobian matrix $\partial\boldsymbol{g}/\partial\boldsymbol{u}$ is much larger than that for $\partial\boldsymbol{f}/\partial\boldsymbol{u}$. There are many practical problems involving the numerical solutions of such equations. An example is the numerical simulation of transient reacting flows, such as the direct numerical simulation (DNS) of the stability and transition of hypersonic reacting boundary-layer flows,[1,2] where high-order-accurate and robust numerical methods are required. The governing equations for transient high-speed reactive flows can be written in the following form:

$$\frac{\partial\boldsymbol{U}}{\partial t} + \frac{\partial\boldsymbol{F}_j}{\partial x_j} = \boldsymbol{W} \tag{2}$$

where $\boldsymbol{F}_j$ are the flux vectors and $\boldsymbol{W}$ is the vector for the reacting source terms. The spatial discretization of Eq. (2) leads to a system of additively split ordinary differential equations (1), where the source term $\boldsymbol{W}$ is taken as the stiff term $\boldsymbol{g}$ and the rest of the convective and diffusive flux terms are taken as the term $\boldsymbol{f}$.

A major computational difficulty lies in the stiffness of temporal integrations of ordinary differential equations after spatial discretizations are applied to the reactive Navier–Stokes equations. The

Received 14 July 2003; revision received 25 March 2004; accepted for publication 29 March 2004. Copyright © 2004 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0001-1452/04 $10.00 in correspondence with the CCC.

*Staff Scientist, Energetic Materials Center, P.O.Box 808, L-268; yoh1@llnl.gov. Member AIAA.

†Professor, Mechanical and Aerospace Engineering Department. Associate Fellow AIAA.

stiffness is introduced from the source term modeling the finite-rate reaction chemistry. To solve Eq. (1), one uses an implicit approach to integrate the equation, so that an excessively small time-step constraint from an explicit approach is avoided. Global implicit methods, however, are computationally expensive for multidimensional reactive flow computations due to the large memory requirements to invert the Jacobian matrix for the implicit terms and prohibitively large amount of computer time. For reactive flow equations where the stiff terms can be additively separated out from the rest [see Eq. (1)], a semi-implicit method can simultaneously treat the nonstiff term $\boldsymbol{f}$ explicitly and the stiff term $\boldsymbol{g}$ implicitly. Examples of the semi-implicit methods include the Adams–Bashforth–Crank–Nicolson (ABCN) method[3–5] often used in the DNS of incompressible turbulent flows and the semi-implicit MacCormack method (see Refs. 6 and 7) used in compressible reactive flow simulation. An alternative method for computing very stiff reactive flow equations is the time-splitting method. This method involves advancing a fraction of time step in solving the stiff equation implicitly and using this predictor step to solve the remaining advection term explicitly.[8–12] The time-splitting method is shown robust for highly stiff equations in many combustion reaction simulations. Nevertheless, its accuracy is well known to degrade to a low-order accuracy at the vicinity of a strong shock.

The main drawback of the previous semi-implicit methods is that their temporal accuracy is usually second order. For the DNS of transient reactive flows, third-order or higher accurate methods are desirable to capture all flow timescales. To obtain high-order-accurate semi-implicit schemes with a good stability property, namely, the L stability, the simultaneous coupling between the explicit and implicit terms needs to be considered in the derivation of the scheme. In Ref. 13, the author derived and analyzed three different versions of semi-implicit Runge–Kutta (SIRK) methods for additively split autonomous differential equations of

$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} = \boldsymbol{f}(\boldsymbol{u}) + \boldsymbol{g}(\boldsymbol{u}) \tag{3}$$

where the nonstiff term $\boldsymbol{f}$ is treated by explicit Runge–Kutta methods and the stiff term $\boldsymbol{g}$ is simultaneously treated by three implicit Runge–Kutta methods. The three implicit methods for $\boldsymbol{g}$ are a diagonally implicit Runge–Kutta method and two linearized Runge–Kutta methods (see Ref. 14).

The SIRK schemes derived in Ref. 13, however, have a drawback in that they are not storage efficient because they require multiple sets of computer memory for each variable in the flowfields. Such computer memory requirement becomes a significant limiting factor in the large-scale computations of multidimensional reactive flows. A number of explicit low-storage Runge–Kutta schemes of third-order accuracy were derived by Williamson,[15] and $2N$ storage version of fully implicit Runge–Kutta scheme is used by Engquist and Sjögreen,[16] where N is the total number of unknown variables. Another drawback of the SIRK schemes derived in Ref. 13 is that they are derived for the autonomous equations (3) only. A classical explicit Runge–Kutta scheme derived for an autonomous equation can easily be extended to solve a nonautonomous equation by means of simple variable transformation. However, a simple extension of an autonomous SIRK to a nonautonomous equation in the form of Eq. (1) is not trivial because of the coupling between the $\boldsymbol{f}$ and $\boldsymbol{g}$ terms in the corresponding order conditions.

More recently, Calvo et al.[17] and Ascher et al.[18] presented the implicit–explicit Runge–Kutta schemes for time-dependent differential equations. By assigning a single constant gamma to represent a diagonal set of implicit coefficients, the authors derived a specialized version of SIRK schemes. Instead, our coefficients of the SIRK schemes presented in this paper are completely determined by the conditions of accuracy and L stability and, thus, more general. Moreover, the schemes require low memory storage and effectively comply with the modern large-scale scientific computing needs.

The biggest challenge in the derivation of the SIRK schemes lies in the search for a set of coefficients satisfying all of the algebraic conditions arising from the accuracy and L-stability requirements. As shown in Ref. 13 for the autonomous equation (3), three-stage methods provide enough free parameters to determine a third-order semi-implicit scheme that is L stable. On the other hand, for the nonautonomous equation (1), there exist extra accuracy conditions related to time-variant terms. Thus, a four-stage SIRK method is needed to reach a third-order accuracy. There is an analogy between the quadrature formulas and the general Runge–Kutta schemes in the derivation of some of the well-known implicit schemes. Butcher compiled a set of simplifying conditions, which provides a controlled guideline to the derivation of high-order implicit Runge–Kutta schemes (see Refs. 19–23). For a third-order additive semi-implicit low-storage method, the number of those simplifying conditions exceeds the general Taylor expansion order conditions. In fact, one's effort to minimize the necessary order conditions for a high-order scheme and maximize additional freedom to choose undetermined coefficients can lead to a successful parameter search. When the current methods need to improve their accuracy of order three to, for example, five or more, a controlled pattern in the order conditions analogous to Butcher's can provide a useful starting point for the search of optimal Runge–Kutta coefficients.

The purpose of this paper is to present new versions of semi-implicit Runge–Kutta schemes, which have a low-storage memory requirement and are suitable for integrating non-autonomous equations for the DNS of reacting flows. In this paper, systematic derivations of all versions of SIRK methods are described for nonautonomous differential systems with new coefficients that replace the coefficients previously reported in Ref. 13. The new third-order, low-storage Runge–Kutta schemes will require only two levels of memory locations during the time integration. We present new methods that are high-order accurate and L stable, suitable for solving Eq. (1). The solution of model equations and multidimensional reacting flows that supports the accuracy and stability of the new schemes is the subject of the companion paper.[24]

In what follows, high-order low-storage SIRK algorithms for integrating Eq. (1) are first presented. The algebraic conditions of accuracy and stability are simultaneously solved to determine the SIRK coefficients. Then, we rederive a set of rational coefficients for the general SIRK schemes from Ref. 13. The two families of low-storage (LS) methods are referred as LSSIRK-4A and LSSIRK-4C for the implicit and linearized implicit methods, respectively. In addition, the general SIRK methods will follow SIRK-3 and SIRK-4 notations for autonomous and nonautonomous methods, respectively. An extensive application of the new algorithms is found in the companion paper.

SIRK Schemes

In numerical computations of reacting flows using the method of lines, spatial derivatives in the governing partial differential equations are first approximated by using a spatial discretization scheme. The spatial discretization leads to a system of first-order differential equations in the form of Eq. (1). For systems of unsteady flow with time-dependent forcing terms or boundary conditions, Eq. (1) is not autonomous, that is, $\boldsymbol{f}(t, \boldsymbol{u})$ and $\boldsymbol{g}(t, \boldsymbol{u})$ are explicit functions of time. For the purpose of using a SIRK scheme to integrate the equation, vector $\boldsymbol{g}(t, \boldsymbol{u})$ contains the stiff source terms and some spatial discretization of the stiff terms. The remaining nonstiff terms are included in vector $\boldsymbol{f}(t, \boldsymbol{u})$. In general, the splitting of $\boldsymbol{f}$ and $\boldsymbol{g}$ terms is not unique, and the SIRK schemes are derived for a given splitting of $\boldsymbol{f}$ and $\boldsymbol{g}$ terms. The splitting of stiff and nonstiff terms in Eq. (1) makes it possible to solve the equation by a SIRK scheme, which is a one-step method involving intermediate stages to achieve high-order accuracy and good stability properties.

A general r-stage SIRK method integrates Eq. (1) by simultaneously treating $\boldsymbol{f}$ explicitly and $\boldsymbol{g}$ implicitly. The simultaneous explicit and implicit treatment in the SIRK method creates a coupling between the explicit and implicit terms. Because of the coupling, the coefficients for the conventional explicit and implicit Runge–Kutta schemes cannot be used in a SIRK calculation. Two versions of SIRK methods are considered in this paper. The first version is denoted as method A (SIRK-rA scheme for an r-stage scheme) in the following form:

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^n + \sum_{j=1}^{r} w_j \boldsymbol{k}_j \tag{4}$$

$$\boldsymbol{k}_i = h\boldsymbol{f}\left(t_n + r_i h, \boldsymbol{u}^n + \sum_{j=1}^{i-1} b_{ij}\boldsymbol{k}_j\right)$$

$$+ h\boldsymbol{g}\left(t_n + s_i h, \boldsymbol{u}^n + \sum_{j=1}^{i-1} c_{ij}\boldsymbol{k}_j + d_i\boldsymbol{k}_i\right) \quad (i = 1, \ldots, r) \tag{5}$$

where h is the time step size. The coefficients can be summarized in a convenient tabular format known as a Butcher array. The two axes contain the weights and time coefficients and the main coefficients are written as a diagonal matrix. The two arrays that summarize the coefficients of the explicit and the implicit parts of SIRK are as follows:

$$\begin{array}{c|cccc}
s_1 & d_1 & & & \\
s_2 & c_{21} & d_2 & & \\
s_3 & c_{31} & c_{32} & d_3 & \\
s_4 & c_{41} & c_{42} & c_{43} & d_4 \\
\hline
 & \omega_1 & \omega_2 & \omega_3 & \omega_4
\end{array}
\qquad
\begin{array}{c|cccc}
r_1 & & & & \\
r_2 & b_{21} & & & \\
r_3 & b_{31} & b_{32} & & \\
r_4 & b_{41} & b_{42} & b_{43} & \\
\hline
 & \omega_1 & \omega_2 & \omega_3 & \omega_4
\end{array}$$

The second version of the method is denoted method C (SIRK-rC scheme). This version is derived by the linearized treatment of the implicit term $\boldsymbol{g}$ as follows:

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^n + \sum_{j=1}^{r} w_j \boldsymbol{k}_j \tag{6}$$

$$\left[\boldsymbol{I} - hd_i\boldsymbol{J}\left(t_n + s_i h, \boldsymbol{u}^n + \sum_{j=1}^{i-1} c_{ij}\boldsymbol{k}_j\right)\right]\boldsymbol{k}_i = h\left[\boldsymbol{f}\left(t_n + r_i h, \boldsymbol{u}^n\right.\right.$$

$$\left.\left. + \sum_{j=1}^{i-1} b_{ij}\boldsymbol{k}_j\right) + \boldsymbol{g}\left(t_n + s_i h, \boldsymbol{u}^n + \sum_{j=1}^{i-1} c_{ij}\boldsymbol{k}_j\right)\right]$$

$$(i = 1, \ldots, r) \tag{7}$$

where $\boldsymbol{J} = \partial\boldsymbol{g}/\partial\boldsymbol{u}$ is the Jacobian matrix of the stiff term $\boldsymbol{g}$.

Accuracy Conditions

In an r-stage SIRK scheme, the coefficients of the scheme are determined in such a way that the resulting scheme meets the requirement for proposed order of accuracy and stability conditions. The accuracy conditions for the undetermined coefficients are obtained by a Taylor expansion. We are mainly interested in deriving the third-order LSSIRK methods for nonautonomous equations. When the system of equations is autonomous in the form of Eq. (3), three-stage SIRK schemes are used for a third-order accuracy. For the nonautonomous equation (1), however, it is found that a four-stage SIRK method is required to satisfy the third-order accuracy.

The following equations are the necessary accuracy equations based on a third-order Taylor expansion. The first 16 equations that hold for all versions of SIRK are

$$\omega_1 + \omega_2 + \omega_3 + \omega_4 = 1, \qquad \omega_2 r_2 + \omega_3 r_3 + \omega_4 r_4 = \tfrac{1}{2}$$

$$\omega_1 s_1 + \omega_2 s_2 + \omega_3 s_3 + \omega_4 s_4 = \tfrac{1}{2}$$

$$\omega_2 b_{21} + \omega_3(b_{31} + b_{32}) + \omega_4(b_{41} + b_{42} + b_{43}) = \tfrac{1}{2}$$

$$\omega_1 d_1 + \omega_2(d_2 + c_{21}) + \omega_3(d_3 + c_{31} + c_{32})$$

$$+ \omega_4(d_4 + c_{41} + c_{42} + c_{43}) = \tfrac{1}{2}$$

$$\omega_2 r_2^2 + \omega_3 r_3^2 + \omega_4 r_4^2 = \tfrac{1}{3}, \qquad \omega_1 s_1^2 + \omega_2 s_2^2 + \omega_3 s_3^2 + \omega_4 s_4^2 = \tfrac{1}{3}$$

$$\omega_1 d_1 s_1 + \omega_2(c_{21}s_1 + d_2 s_2) + \omega_3(c_{31}s_1 + c_{32}s_2 + d_3 s_3)$$

$$+ \omega_4(c_{41}s_1 + c_{42}s_2 + c_{43}s_3 + d_4 s_4) = \tfrac{1}{6}$$

$$\omega_1 d_1^2 + \omega_2[c_{21}d_1 + d_2(c_{21} + d_2)] + \omega_3[d_1 c_{31} + c_{32}(d_2 + c_{21})$$

$$+ d_3(c_{31} + c_{32} + d_3)] + \omega_4[c_{41}d_1 + c_{42}(d_2 + c_{21})$$

$$+ c_{43}(c_{31} + c_{32} + d_3) + d_4(c_{41} + c_{42} + c_{43} + d_4)] = \tfrac{1}{6}$$

$$\omega_3 b_{32} r_2 + \omega_4(b_{42} r_2 + b_{43} r_3) = \tfrac{1}{6}$$

$$\omega_2 b_{21}^2 + \omega_3(b_{31} + b_{32})^2 + \omega_4(b_{41} + b_{42} + b_{43})^2 = \tfrac{1}{3}$$

$$\omega_2 r_2 b_{21} + \omega_3 r_3(b_{31} + b_{32}) + \omega_4 r_4(b_{41} + b_{42} + b_{43}) = \tfrac{1}{3}$$

$$\omega_3 b_{32} b_{21} + \omega_4[b_{42} b_{21} + b_{43}(b_{31} + b_{32})] = \tfrac{1}{6}$$

$$\omega_2(b_{21}d_2 + b_{21}d_1) + \omega_3(d_1 b_{31} + d_2 b_{32} + c_{21}b_{32} + b_{21}c_{32}$$

$$+ d_3 b_{31} + d_3 b_{32}) + \omega_4[b_{41}d_1 + b_{42}(d_2 + c_{21}) + c_{42}b_{21}$$

$$+ b_{43}(c_{31} + c_{32} + d_3) + c_{43}(b_{31} + b_{32})$$

$$+ d_4(b_{41} + b_{42} + b_{43})] = \tfrac{1}{3}$$

$$\omega_2 d_2 r_2 + \omega_3(c_{32}r_2 + d_3 r_3) + \omega_4(c_{42}r_2 + c_{43}r_3 + d_4 r_4) = \tfrac{1}{6}$$

$$\omega_2 b_{21}s_1 + \omega_3(b_{31}s_1 + b_{32}s_2) + \omega_4(b_{41}s_1 + b_{42}s_2 + s_{43}s_3) = \tfrac{1}{6} \tag{8}$$

The remaining two accuracy equations differ for each version of the semi-implicit schemes. For method A,

$$\omega_1 d_1^2 + \omega_2(c_{21} + d_2)^2 + \omega_3(c_{31} + c_{32} + d_3)^2$$

$$+ \omega_4(c_{41} + c_{42} + c_{43} + d_4)^2 = \tfrac{1}{3} \tag{9}$$

$$\omega_1 d_1 s_1 + \omega_2 s_2(c_{21} + d_2) + \omega_3 s_3(c_{31} + c_{32} + d_3)$$

$$+ \omega_4 s_4(c_{41} + c_{42} + c_{43} + d_4) = \tfrac{1}{3} \tag{10}$$

and for method C,

$$\omega_2\left(c_{21}^2 + 2d_2 c_{21}\right) + \omega_3\left[(c_{31} + c_{32})^2 + 2d_3(c_{31} + c_{32})\right]$$

$$+ \omega_4\left[(c_{41} + c_{42} + c_{43})^2 + 2d_4(c_{41} + c_{42} + c_{43})\right] = \tfrac{1}{3} \tag{11}$$

$$\omega_2 s_2(c_{21} + d_2) + \omega_3 s_3(c_{31} + c_{32} + d_3)$$

$$+ \omega_4 s_4(c_{41} + c_{42} + c_{43} + d_4) = \tfrac{1}{3} \tag{12}$$

There are total 18 accuracy equations involved in developing a four-stage SIRK scheme for nonautonomous equation (1). Consequently, the SIRK coefficients will be determined based on these accuracy conditions and a stability condition that is discussed next. The accuracy conditions suggest the following standard assumptions for Runge–Kutta schemes:

$$s_i = \sum_{j=1}^{i-1} c_{ij} + d_i \tag{13}$$

$$r_i = \sum_{j=1}^{i-1} b_{ij} \tag{14}$$

Then, the 14th condition in Eq. (8) is a mere summation of the 15th and 16th conditions. A conventional summation form may be found in the Appendix.

Linear Stability Condition

In addition to accuracy conditions, the third-order SIRK schemes are required to satisfy a stability condition for methods to be L stable for the stiff term $\boldsymbol{g}$. Because of a coupling between the explicit and the implicit terms of semi-implicit computations, the linear stability analysis of this section is slightly different from the standard context.[20] The linear stability analysis is considered with the following scalar model equation:

$$\frac{\mathrm{d}u}{\mathrm{d}t} = \lambda_f u + \lambda_g u \tag{15}$$

where λ_f and λ_g represent the eigenvalues of $\partial \boldsymbol{f}/\partial \boldsymbol{u}$ and $\partial \boldsymbol{g}/\partial \boldsymbol{u}$, respectively, for the split ordinary differential equation (1). Substituting Eq. (15) into any of the SIRK methods leads to the following equation for a characteristic root γ:

$$\gamma = \frac{u^{n+1}}{u^n} = 1 + \sum_{j=1}^{r} \omega_j k_j \tag{16}$$

$$k_i = \left[h\lambda_f\left(1 + \sum_{j=1}^{i-1} b_{ij}k_j\right) + h\lambda_g\left(1 + \sum_{j=1}^{i-1} c_{ij}k_j\right)\right] \Bigg/ (1 - d_i h\lambda_g) \qquad (i = 1, \dots, r) \tag{17}$$

where γ is a function of $h\lambda_f$ and $h\lambda_g$.

An $A(\alpha)$ stability region of a semi-implicit method in the complex plane of $h\lambda_f$ is defined as the region where

$$|\gamma\{h\lambda_f, h\lambda_g\}| \le 1 \tag{18}$$

for the left-half complex domain of $h\lambda_f$ and for all $h\lambda_g$ within a wedge bounded by $[\pi - \alpha, \pi + \alpha]$ in the complex plane. When $\alpha = \pi/2$, the split SIRK method is defined as A stable. Furthermore, the L-stable condition is derived by letting $|\gamma\{h\lambda_f, h\lambda_g\}| \to 0$, while $Re(h\lambda_g) \to -\infty$. The L-stability requirement of a SIRK scheme results in an additional algebraic condition to accuracy, namely,

$$1 + \sum_{j=1}^{r} \omega_j \beta_j = 0 \tag{19}$$

with

$$\beta_i = -\frac{1}{d_i}\left[1 + \sum_{j=1}^{i-1} c_{ij}\beta_j\right] \qquad (i = 1, \dots, r) \tag{20}$$

LSSIRK Schemes for Nonautonomous Equations

LS versions of the third-order, four-stage SIRK schemes are derived in this section. A general four-stage semi-implicit scheme requires the storage of variables at $5N$ locations. Such storage requirement is often too large for multidimensional reactive flow computations. A set of third-order SIRK schemes that requires no more than $2N$ storage locations is derived so that they simultaneously satisfy accuracy conditions and the L-stability condition. In this paper, only two versions of the third-order LSSIRK schemes (methods A and C) are derived. They are denoted as LSSIRK-4A and LSSIRK-4C schemes, respectively.

For a nonsplit differential equation, a traditional Runge–Kutta scheme derived for an autonomous equation, $y' = f(u)$, can be extended to a nonautonomous equation, $y' = f(t, u)$, by a simple transformation. No new coefficients are needed to solve the nonautonomous equation. When the governing ordinary differential equation is a split nonautonomous equation (1), however, the SIRK schemes derived for the autonomous equation (3) can not be applied to the nonautonomous equation by a similar transformation because of a coupling between explicit and implicit terms. A new set of coefficients are required in the nonautonomous case.

The third-order LSSIRK-4A scheme can be expressed as

$$\begin{gathered}
\boldsymbol{k}_1 = h[\boldsymbol{f}(t_0, \boldsymbol{u}_0) + \boldsymbol{g}(t_0 + s_1 h, \boldsymbol{u}_0 + c_1\boldsymbol{k}_1)] \\
\boldsymbol{u}_1 = \boldsymbol{u}_0 + b_1\boldsymbol{k}_1 \\
\boldsymbol{k}_2 = a_2\boldsymbol{k}_1 + h[\boldsymbol{f}(t_0 + r_2 h, \boldsymbol{u}_1) + \boldsymbol{g}(t_0 + s_2 h, \boldsymbol{u}_1 + \bar{c}_2\boldsymbol{k}_1 + c_2\boldsymbol{k}_2)] \\
\boldsymbol{u}_2 = \boldsymbol{u}_1 + b_2\boldsymbol{k}_2 \\
\boldsymbol{k}_3 = a_3\boldsymbol{k}_2 + h[\boldsymbol{f}(t_0 + r_3 h, \boldsymbol{u}_2) + \boldsymbol{g}(t_0 + s_3 h, \boldsymbol{u}_2 + \bar{c}_3\boldsymbol{k}_2 + c_3\boldsymbol{k}_3)] \\
\boldsymbol{u}_3 = \boldsymbol{u}_2 + b_3\boldsymbol{k}_3 \\
\boldsymbol{k}_4 = a_4\boldsymbol{k}_3 + h[\boldsymbol{f}(t_0 + r_4 h, \boldsymbol{u}_3) + \boldsymbol{g}(t_0 + s_4 h, \boldsymbol{u}_3 + \bar{c}_4\boldsymbol{k}_3 + c_4\boldsymbol{k}_4)] \\
\boldsymbol{u}_4 = \boldsymbol{u}_3 + b_4\boldsymbol{k}_4
\end{gathered} \tag{21}$$

Similarly, the third-order LSSIRK-4C scheme for nonautonomous equations is expressed as

$$\begin{aligned}
&[\boldsymbol{I} - hc_1\boldsymbol{J}(t_0 + s_1 h, \boldsymbol{u}_0)]\boldsymbol{k}_1 = h[\boldsymbol{f}(t_0, \boldsymbol{u}_0) + \boldsymbol{g}(t_0 + s_1 h, \boldsymbol{u}_0)] \\
&\boldsymbol{u}_1 = \boldsymbol{u}_0 + b_1\boldsymbol{k}_1 \\
&[\boldsymbol{I} - hc_2\boldsymbol{J}(t_0 + s_2 h, \boldsymbol{u}_1 + \bar{c}_2\boldsymbol{k}_1)]\boldsymbol{k}_2 = h[\boldsymbol{f}(t_0 + r_2 h, \boldsymbol{u}_1) \\
&\qquad + \boldsymbol{g}(t_0 + s_2 h, \boldsymbol{u}_1 + \bar{c}_2\boldsymbol{k}_1)] + a_2[\boldsymbol{I} - hc_2\boldsymbol{J}(t_0 + s_2 h, \boldsymbol{u}_1 + \bar{c}_2\boldsymbol{k}_1)]\boldsymbol{k}_1 \\
&\boldsymbol{u}_2 = \boldsymbol{u}_1 + b_2\boldsymbol{k}_2 \\
&[\boldsymbol{I} - hc_3\boldsymbol{J}(t_0 + s_3 h, \boldsymbol{u}_2 + \bar{c}_3\boldsymbol{k}_2)]\boldsymbol{k}_3 = h[\boldsymbol{f}(t_0 + r_3 h, \boldsymbol{u}_2) \\
&\qquad + \boldsymbol{g}(t_0 + s_3 h, \boldsymbol{u}_2 + \bar{c}_3\boldsymbol{k}_2)] + a_3[\boldsymbol{I} - hc_3\boldsymbol{J}(t_0 + s_3 h, \boldsymbol{u}_2 + \bar{c}_3\boldsymbol{k}_2)]\boldsymbol{k}_2 \\
&\boldsymbol{u}_3 = \boldsymbol{u}_2 + b_3\boldsymbol{k}_3 \\
&[\boldsymbol{I} - hc_4\boldsymbol{J}(t_0 + s_4 h, \boldsymbol{u}_3 + \bar{c}_4\boldsymbol{k}_3)]\boldsymbol{k}_4 = h[\boldsymbol{f}(t_0 + r_4 h, \boldsymbol{u}_3) \\
&\qquad + \boldsymbol{g}(t_0 + s_4 h, \boldsymbol{u}_3 + \bar{c}_4\boldsymbol{k}_3)] + a_4[\boldsymbol{I} - hc_4\boldsymbol{J}(t_0 + s_4 h, \boldsymbol{u}_3 + \bar{c}_4\boldsymbol{k}_3)]\boldsymbol{k}_3 \\
&\boldsymbol{u}_4 = \boldsymbol{u}_3 + b_4\boldsymbol{k}_4
\end{aligned} \tag{22}$$

Unlike the general r-stage Runge–Kutta methods that have rN storage requirements for one step advancement, the LS Runge–Kutta scheme requires only memory locations for $\boldsymbol{k}_j$ and $\boldsymbol{u}_j$ at each of the general r stage.

The LSSIRK schemes just presented can be written in terms of the standard SIRK coefficients. In other words, there are 20 transformation equations between the SIRK and LSSIRK for nonautonomous equations. The following 14 transformation equations are the same for both LSSIRK-4A and LSSIRK-4C schemes:

$$\begin{gathered}
\omega_1 = b_1 + b_2a_2 + b_3a_3a_2 + b_4a_4a_3a_2, \qquad \omega_2 = b_2 + b_3a_3b_4a_4a_3 \\
\omega_3 = b_3 + b_4a_4, \qquad \omega_4 = b_4 \\
b_{21} = b_1, \qquad b_{31} = b_1 + b_2a_2, \qquad b_{32} = b_2 \\
b_{41} = b_1 + b_2a_2 + b_3a_3a_2, \qquad b_{42} = b_2 + b_3a_3, \qquad b_{43} = b_3 \\
d_1 = c_1, \qquad d_2 = c_2, \qquad d_3 = c_3, \qquad d_4 = c_4
\end{gathered} \tag{23}$$

The remaining six transformation equations are different for LSSIRK method A (LSSIRK-4A) and method C (LSSIRK-4C) schemes. For the LSSIRK-4A scheme, the additional relations are

$$\begin{gathered}
c_{21} = b_1 + \bar{c}_2 + c_2a_2, \qquad c_{31} = b_1 + b_2a_2 + \bar{c}_3a_2 + a_2c_3a_3 \\
c_{32} = b_2 + \bar{c}_3 + c_3a_3 \\
c_{41} = b_1 + b_2a_2 + (b_3 + \bar{c}_4)a_3a_2 + c_4a_4a_3a_2 \\
c_{42} = b_2 + (b_3 + \bar{c}_4)a_3 + c_4a_4a_3 \\
c_{43} = b_3 + \bar{c}_4 + c_4a_4
\end{gathered} \tag{24}$$

The additional relations for the LSSIRK-4C method are

$$\begin{gathered}
c_{21} = b_1 + \bar{c}_2, \qquad c_{31} = b_1 + b_2a_2 + \bar{c}_3a_2 \\
c_{32} = b_2 + \bar{c}_3, \qquad c_{41} = b_1 + b_2a_2 + (b_3 + \bar{c}_4)a_3a_2 \\
c_{42} = b_2 + (b_3 + \bar{c}_4)a_3, \qquad c_{43} = b_3 + \bar{c}_4
\end{gathered} \tag{25}$$

The coefficients for the third-order LSSIRK schemes are determined by solving the 18 accuracy equations given by Eqs. (8–12). These accuracy equations are solved for the unknown coefficients. We then solve these algebraic equations with three predefined coefficients of our choice because there are a total of 22 unknowns for 19 equations, that is, 18 accuracy and 1 L stability. For the LSSIRK-4A scheme, the derived coefficients are both third-order accurate and L stable. We also learn that it is more difficult to determine LSSIRK-4C coefficients with all 19 conditions. Instead, only accuracy conditions are solved for the LSSIRK-4C coefficients. Because there are many possible sets of solutions for the system of linear equations, we can find one set of rational numbers in that coefficients are reasonably ranged for numerical implementation. For the LSSIRK-4A scheme, a set of coefficients in rational numbers can be found by solving the accuracy and stability equations. The coefficients for L-stable, third-order method-A (LSSIRK-4A) scheme are

$$b_1 = 3/4, \qquad b_2 = -2/27, \qquad b_3 = 2, \qquad b_4 = 2/3$$

$$a_2 = 23/4, \qquad a_3 = -1/9, \qquad a_4 = -5/2$$

$$r_2 = 3/4, \qquad r_3 = 1/4, \qquad r_4 = 3/4$$

$$s_1 = 2, \qquad s_2 = 79/28, \qquad s_3 = 127/84, \qquad s_4 = 11/84$$

$$c_1 = 2, \qquad c_2 = 10{,}901/12{,}096, \qquad c_3 = 7601/1344$$

$$c_4 = 3/4, \qquad \bar{c}_2 = -1027/256$$

$$\bar{c}_3 = -817/36{,}288, \qquad \bar{c}_4 = -605/168$$

For method C, abscissas of implicit time integration, namely, s_i, are set equal to their explicit counterparts, r_i. The coefficients for the non L-stable LSSIRK-4C scheme are

$$b_1 = 1/3, \qquad b_2 = -30/73, \qquad b_3 = 11/150, \qquad b_4 = -25/4$$

$$a_2 = -17/90, \qquad a_3 = 1710/803, \qquad a_4 = -517/1250$$

$$r_1 = 0, \qquad r_2 = 1/3, \qquad r_3 = 0, \qquad r_4 = 1/5$$

$$\bar{c}_3 = -1/2, \qquad \bar{c}_2 = -0.189210, \qquad \bar{c}_4 = 0.776465$$

$$c_1 = 0.102712, \qquad c_2 = 0.531583$$

$$c_3 = 0.881764, \qquad c_4 = 0.776465$$

SIRK Schemes with Rational Coefficients

In Ref. 13, three sets of SIRK methods for the split autonomous ordinary differential equations (3) were derived, where the combination of explicit and implicit Runge–Kutta schemes are used to treat the nonstiff and stiff parts of the equation. For a third-order accuracy, eight accuracy conditions of the coupled treatment of $\boldsymbol{f}(u)$ and $\boldsymbol{g}(u)$ are derived. An additional L-stability condition further allows for multiple families of Runge–Kutta coefficients with three free parameters. In Ref. 13, only the coefficients in finite decimals were obtained by numerically solving the accuracy and stability equations. In applications, it is easier to control numerical accuracy of the coefficients of the SIRK schemes if the coefficients are rational numbers or can be computed by closed-form analytical formulas.

In this section, the coefficients for the three versions of the third-order SIRK schemes of Ref. 13 are rederived in rational numbers. They are derived by requiring to satisfy the accuracy and stability conditions, along with the requirement that implicit coefficients are positive. The algebraic conditions for the autonomous equations are

$$\omega_1 + \omega_2 + \omega_3 = 1, \qquad \omega_2 b_{21} + \omega_3(b_{31} + b_{32}) = \tfrac{1}{2}$$

$$\omega_1 d_1 + \omega_2(d_2 + c_{21}) + \omega_3(d_3 + c_{31} + c_{32}) = \tfrac{1}{2}$$

$$\omega_1 d_1^2 + \omega_2[c_{21}d_1 + d_2(c_{21} + d_2)] + \omega_3[d_1 c_{31} + c_{32}(d_2 + c_{21}) + d_3(c_{31} + c_{32} + d_3)] = \tfrac{1}{6}$$

$$\omega_2 b_{21}^2 + \omega_3(b_{31} + b_{32})^2 = \tfrac{1}{3}, \qquad \omega_3 b_{32} b_{21} = \tfrac{1}{6}$$

$$\omega_2(b_{21}d_2 + b_{21}d_1) + \omega_3(d_1 b_{31} + d_2 b_{32} + c_{21}b_{32} + b_{21}c_{32} + d_3 b_{31} + d_3 b_{32}) = \tfrac{1}{3} \qquad (26)$$

The remaining last condition is different for the three versions of SIRK schemes.

Method A:

$$\omega_1 d_1^2 + \omega_2(c_{21} + d_2)^2 + \omega_3(c_{31} + c_{32} + d_3)^2 = \tfrac{1}{3} \qquad (27)$$

Method B:

$$\omega_2 c_{21}^2 + \omega_3(c_{31} + c_{32})^2 = \tfrac{1}{3} \qquad (28)$$

Method C:

$$\omega_2\left(c_{21}^2 + 2d_2 c_{21}\right) + \omega_3\left[(c_{31} + c_{32})^2 + 2d_3(c_{31} + c_{32})\right] = \tfrac{1}{3} \qquad (29)$$

The eight accuracy conditions listed [Eq. (26) and either Eq. (27), (28), or (29)] and the additional stability condition given by Eq. (18) are solved for 12 undetermined coefficients. There are three free parameters. We choose $w_1 = \frac{1}{8}$, $d_1 = \frac{3}{4}$, and $w_2 = \frac{1}{8}$. By considerable manipulations on the remaining algebraic equations, nine rational coefficients are found. What differs in our approach from Ref. 13 is that, in this work, we have concentrated on finding the analytical form of the roots by grouping approaches. As a result, three sets of third-order SIRK coefficients have been derived in rational numbers. As an example, a set of SIRK-3A coefficients consists of the following;

$$w_1 = 1/8, \qquad w_2 = 1/8, \qquad w_3 = 3/4$$

$$b_{21} = 8/7, \qquad b_{31} = 71/252, \qquad b_{32} = 7/36$$

$$d_1 = 3/4, \qquad d_2 = 75/233, \qquad d_3 = 65/168$$

$$c_{21} = 5589/6524, \qquad c_{31} = 7691/26{,}096$$

$$c_{32} = -26{,}335/78{,}288$$

The coefficients for 3B and 3C are listed in the Appendix.

For the nonautonomous split differential equation (1), four-stage methods are required to achieve a third-order accuracy. Because four stages are used, we denote new schemes as SIRK-4A and SIRK-4C. The accuracy conditions of Eqs. (8–12) are solved for the unknown coefficients. If there is no rational coefficient, we use a decimal number that is found by the numerical root solver, instead.

For the SIRK-4A scheme, the following coefficients are derived to satisfy both the accuracy and the L-stability conditions. Their coefficients are

$$\omega_1 = 13/100, \qquad \omega_2 = 1/4, \qquad \omega_3 = 13/25, \qquad \omega_4 = 1/10$$

$$b_{21} = 0.338170, \qquad b_{31} = -0.019088, \qquad b_{32} = 0.779584$$

$$b_{41} = -3/10, \qquad b_{42} = 1/5, \qquad b_{43} = 3/10$$

$$c_{21} = -147/500, \qquad c_{31} = 0.149135, \qquad c_{32} = 1/5$$

$$c_{41} = -1.13081, \qquad c_{42} = 1.78081, \qquad c_{43} = -1/2$$

$$d_1 = 117{,}481/100{,}000, \qquad d_2 = 0.526767$$

$$d_3 = 0.158717, \qquad d_4 = 1/10$$

where

$$r_i = \sum_{j=1}^{i-1} b_{ij}, \qquad s_i = d_i + \sum_{j=1}^{i-1} c_{ij}$$

The coefficients of a companion scheme, SIRK-4C, are listed in the Appendix.

$A(\alpha)$ Stability Regions of SIRK Schemes

In the preceding section, three groups of new third-order SIRK schemes have been developed with various properties. They are 1) SIRK-3A, SIRK-3B, and SIRK-3C methods for the autonomous equation; 2) SIRK-4A and SIRK-4C methods for the nonautonomous equation; and 3) LSSIRK-4A and LSSIRK-4C, LSSIRK methods for the nonautonomous equation.

In the case of a classical explicit Runge–Kutta scheme for a single ordinary differential equation in the form of $u' = f(t, u)$, the stability is evaluated by applying a linear model equation, $u' = \lambda u$, where λ is a complex number with nonpositive real part. The region of stability is defined as the region in the $h\lambda$ complex plane, where the magnitude of the characteristic root of the scheme is not greater than 1. In this section, we calculate the stability region of the SIRK schemes based on the scalar split model equation of Eq. (15). An $A(\alpha)$ stability region of semi-implicit methods is computed in the complex plane of $h\lambda_f$ by Eq. (18). In other words, a SIRK is L stable to the stiff term $h\lambda_g$, whereas there is a limited stability region for the nonstiff term $h\lambda_f$ because the nonstiff term f is treated explicitly.

Figure 1 shows the region of $A(\alpha)$ stability for the explicit term $h\lambda_f$ for all possible $h\lambda_g$ within a wedge bounded by $[\pi/2, 3\pi/2]$ in the complex plane for the SIRK-3C scheme. Figure 1 shows that, when $\alpha = 0$, the stability region in $h\lambda_f$ is similar to a classical explicit third-order Runge–Kutta (RK) scheme. The stability of the semi-implicit scheme is not affected by the implicit treatment of the stiff term. As the wedge angle α for the implicit term increases from 0 to 90 deg, the stability region for $h\lambda_f$ of the SIRK-3C scheme decreases somewhat. The SIRK-3A and SIRK-3B schemes are also found to have similar stability regions, which are not shown here.

The second group of the SIRK schemes are the four-stage third-order SIRK schemes for the nonautonomous equation (1). The third-order RK schemes for nonautonomous equations require four stages (SIRK-4A and SIRK-4C) because additional accuracy conditions are needed to achieve high order. The $A(\alpha)$ stability regions of SIRK-4A and SIRK-4C schemes for nonautonomous equations are shown in Figs. 2 and 3, respectively. Figures 2 and 3 show that the SIRK-4A method retains similar stability properties as those of the SIRK-3C method. The stability region of the SIRK-4C method is good when α is less than 67.5 deg. This condition is satisfied for most of the stiff differential equation. For the case of α close to 90 deg, the stability region is more restricted because the L-stability condition is not imposed in the SIRK-4C derivation.

The third group of the SIRK schemes is the third-order low-storage schemes. The $A(\alpha)$ stability regions of third-order low-storage RK schemes (LSSIRK-4A and LSSIRK-4C schemes) are computed for the nonautonomous equation. Figures 4 and 5 show the regions of stability for the explicit term $h\lambda_f$ for all possible $h\lambda_g$ within a wedge bounded by $[\pi/2, 3\pi/2]$ in the complex plane. Figures 4 and 5 show that the LSSIRK-4A scheme retains similar stability properties as those of the SIRK-3C method. Like the case of the SIRK-4C method, the LSSIRK-4C method is derived to satisfy the accuracy equations only. Therefore, the stability region of LSSIRK-4C is more restrictive when α approaches 90 deg. Therefore, for stiff equations, the L-stable LSSIRK-4A is recommended over the LSSIRK-4C method.

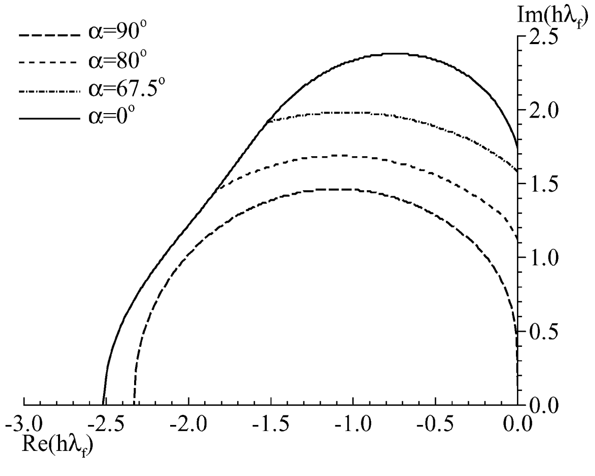

Fig. 1 **$A(\alpha)$-stability region of the SIRK-3C method for the explicit terms $h\lambda_f$.**

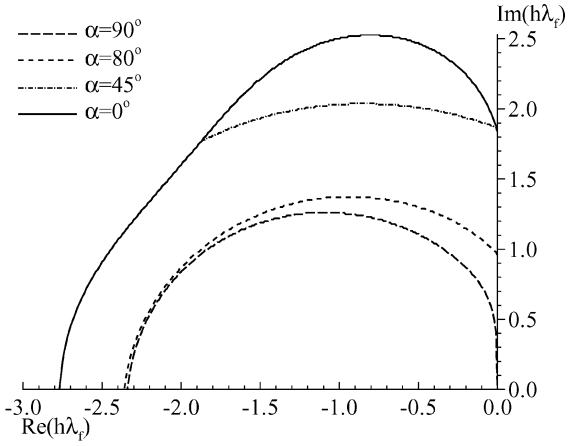

Fig. 2 **$A(\alpha)$-stability region of the SIRK-4A method for the explicit terms $h\lambda_f$.**

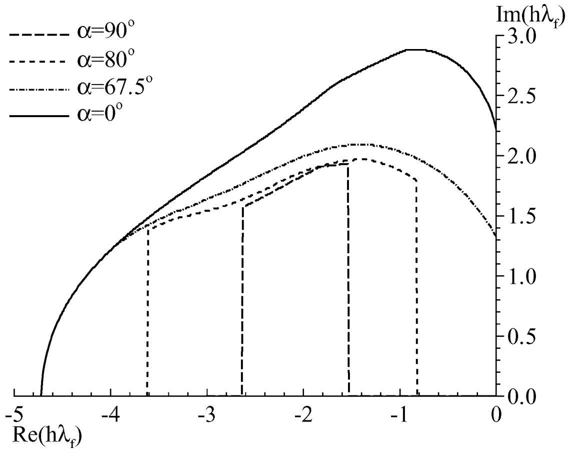

Fig. 3 **$A(\alpha)$-stability region of the SIRK-4C method for the explicit terms $h\lambda_f$.**

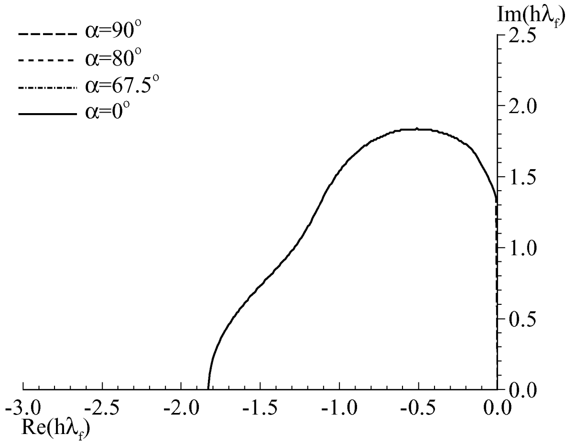

Fig. 4 **$A(\alpha)$-stability region of the LSSIRK-4A method for the explicit terms $h\lambda_f$.**

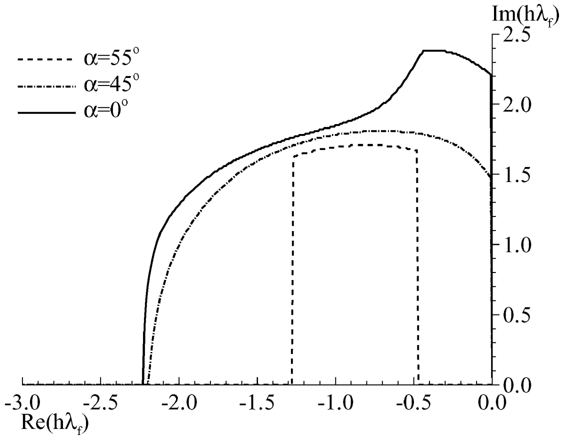

Fig. 5 $A(\alpha)$**-stability region of the LSSIRK-4C method for the explicit terms** $h\lambda_f$**.**

Conclusions

New high-order, semi-implicit RK schemes are developed to simulate transient reacting flows. Because the high-order L-stable implicit method treats $g(t, u)$ and high-order explicit method treats $f(t, u)$ independently, the cost of running the hybrid scheme is only a fraction of that needed to do a full implicit calculation. At the same time, the efficiency of the explicit RK method is maintained through out the simultaneous coupling of the two. The scheme is guaranteed to be both high-order accurate and L stable. The coupling between implicit and explicit methods, furthermore, removes the prohibitively small time step requirement of solving stiff differential equations. Because of its LS framework, the present scheme is also suitable for large-scale computations involving a massive memory requirement.

Appendix: Summation Form and Coefficients

Accuracy Conditions in Summation Form

We construct general accuracy conditions (8–12) in summation form. As before, the first 16 are general, and the 17th and 18th differ for method A and method C:

$$\sum_i \omega_i = 1, \qquad \sum_i \omega_i r_i = \frac{1}{2}, \qquad \sum_i \omega_i s_i = \frac{1}{2}$$

$$\sum_i \omega_i \sum_j^i b_{ij} = \frac{1}{2}, \qquad \sum_i \omega_i \sum_j^i c_{ij} = \frac{1}{2}, \qquad \sum_i \omega_i r_i^2 = \frac{1}{3}$$

$$\sum_i \omega_i s_i^2 = \frac{1}{3}, \qquad \sum_i \omega_i \sum_j^i c_{ij} s_j = \frac{1}{6}$$

$$\sum_i \omega_i \sum_j^i c_{ij} \sum_k^j c_{jk} = \frac{1}{6}, \qquad \sum_i \omega_i \sum_j^i b_{ij} r_j = \frac{1}{6}$$

$$\sum_i \omega_i \left(\sum_j^i b_{ij} \right)^2 = \frac{1}{3}, \qquad \sum_i \omega_i r_i \sum_j^i b_{ij} = \frac{1}{3}$$

$$\sum_i \omega_i \sum_j^i b_{ij} \sum_k^j b_{jk} = \frac{1}{6}$$

$$\sum_i \omega_i \left(\sum_j^i c_{ij} \sum_k^j b_{jk} + \sum_j^i b_{ij} \sum_k^j c_{jk} \right) = \frac{1}{3}$$

$$\sum_i \omega_i \sum_j^i c_{ij} r_j = \frac{1}{6}, \qquad \sum_i \omega_i \sum_j^i b_{ij} s_j = \frac{1}{6} \tag{A1}$$

Two additional conditions for method A are

$$\sum_i \omega_i \left(\sum_j^i c_{ij} \right)^2 = \frac{1}{3}, \qquad \sum_i \omega_i s_i \sum_j^i c_{ij} = \frac{1}{3} \tag{A2}$$

and for method C, we have

$$\sum_i \omega_i \sum_j^{i-1} c_{ij} \sum_j^i (c_{ij} + c_{ii}) = \frac{1}{3}, \qquad \sum_i \omega_i s_i \sum_j^i c_{ij} = \frac{1}{3} \tag{A3}$$

Coefficients in 16-Decimal Approximation

The listed coefficients are either rational numbers or 16-decimal approximations.

SIRK-3B (L stable, autonomous):

$$d_1 = 1.403160446775581, \qquad d_2 = 0.3222947153259484$$

$$d_3 = 0.3153416455775987$$

$$c_{21} = 1.560563684998894, \qquad c_{31} = \tfrac{1}{2}$$

$$c_{32} = -0.6963447867610024$$

SIRK-3C (L stable, autonomous):

$$d_1 = 0.7970967740096232, \qquad d_2 = 0.5913813968007854$$

$$d_3 = 0.1347052663841181$$

$$c_{21} = 1.058925354610082, \qquad c_{31} = \tfrac{1}{2}$$

$$c_{32} = -0.3759391872875334$$

SIRK-4C (not L stable, nonautonomous):

$$\omega_1 = 1/8, \qquad \omega_2 = 1/4, \qquad \omega_3 = 21/40, \qquad \omega_4 = 1/10$$

$$b_{21} = 0.3299167710731796, \qquad b_{31} = -0.003584629502199719$$

$$b_{32} = 0.7626718813721142, \qquad b_{41} = 3/10$$

$$b_{42} = -1, \qquad b_{43} = 89/100$$

$$c_{21} = 3/20, \qquad c_{31} = 8409/250{,}000$$

$$c_{32} = 0.7116738279305653, \qquad c_{41} = 314{,}661/1{,}000{,}000$$

$$c_{42} = -1.253976571187243, \qquad c_{43} = 0.7553162838891784$$

$$d_1 = 0.2171130238473288, \qquad d_2 = 0.0918145303512467$$

$$d_3 = 41{,}351/1{,}000{,}000, \qquad d_4 = 0.1781023349753196$$

Acknowledgments

This research was supported by the Air Force Office of Scientific Research under Grant F49620-97-1-0030. The work was performed under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livermore National Laboratory, under Contract W-7405-Eng-48.

References

[1] Kleiser, L., and Zang, T. A., "Numerical Simulation of Transition in Wall-Bounded Shear Flows," *Annual Review of Fluid Mechanics*, Vol. 23, 1991, pp. 495–535.

[2] Zhong, X., "Direct Numerical Simulation of Hypersonic Boundary Layer Transition Over Blunt Leading Edges, Part I: A New Numerical Method and Validation," AIAA Paper 97-0755, 1997.

[3] Canuto, C., Hussaini, M. Y., Quarteroni, A., and Zang, T. A., *Spectral Methods in Fluid Dynamics*, Springer-Verlag, New York, 1988.

[4] Hairer, E., and Wanner, G., *Solving Ordinary Differential Equations II*, Springer-Verlag, Berlin, 1991.

[5] Hairer, E., Norsett, S. P., and Wanner, G., *Solving Ordinary Differential Equations I*, Springer-Verlag, Berlin, 1993.

[6] Yee, H. C., and Shin, J. L., "Semi-Implicit and Fully Implicit Shock-Capturing Methods for Nonequilibrium Flows," *AIAA Journal*, Vol. 27, 1989, pp. 299–307.

[7] LeVeque, R. J., and Yee, H. C., "A Study of Numerical Methods for Hyperbolic Conservation Laws with Stiff Source Terms," *Journal of Computational Physics*, Vol. 86, 1990, pp. 187–210.

[8] Yanenko, N. N., *The Method of Fractional Steps*, Springer-Verlag, Berlin, 1971.

[9] Oran, E. S., and Boris, J. P., *Numerical Simulation of Reactive Flow*, Elsevier Science, New York, 1987.

[10] Strang, G., "On the Construction and Comparison of Difference Schemes," *SIAM Journal on Numerical Analysis*, Vol. 5, 1968, pp. 506–517.

[11] Jin, S., "Runge–Kutta Methods for Hyperbolic Conservation Laws with Stiff Relaxation Terms," *Journal of Computational Physics*, Vol. 122, 1995, pp. 51–67.

[12] Rouhi, A., and Wright, J., "A New Operator Splitting Method for the Numerical Solution of Partial Differential Equations," *Computational Physics Communications*, Vol. 85, 1995, pp. 18–28.

[13] Zhong, X., "Additive Semi-implicit Runge–Kutta Methods for Computing High-Speed Nonequilibrium Reactive Flows," *Journal of Computational Physics*, Vol. 128, 1996, pp. 19–31.

[14] Rosenbrock, H. H., "Some General Implicit Processes for the Numerical Solution of Differential Equations," *Computer Journal*, Vol. 5, 1963, pp. 329–331.

[15] Williamson, J. H., "Low-Storage Runge–Kutta Schemes," *Journal of Computational Physics*, Vol. 35, 1995, pp. 48–56.

[16] Engquist, B. E., and Sjögreen, B., "Robust Difference Approximations of Stiff Inviscid Detonation Waves," Computational and Applied Mathematics, CAM Rept. 91-03, Univ. of California, Los Angeles, March 1991.

[17] Calvo, M. P., de Frutos, J., and Novo, J., "Linearly Implicit Runge–Kutta Methods for Advection–Reaction–Diffusion Equations," *Applied Numerical Mathematics*, Vol. 37, 2001, pp. 535–549.

[18] Ascher, U. M., Ruuth, S. J., and Spiteri, R. J., "Implicit–Explicit Runge–Kutta Methods for Time-Dependent Partial Differential Equations," *Applied Numerical Mathematics*, Vol. 27, 1997, pp. 151–167.

[19] Dekker, K., and Verwer, J. G., *Stability of Runge–Kutta Methods for Stiff Nonlinear Differential Equations*, Elsevier Science B.V., Amsterdam, 1984.

[20] Lambert, J. D., *Numerical Methods for Ordinary Differential Systems*, Wiley, New York, 1991.

[21] Lambert, J. D., *Computational Methods in Ordinary Differential Equations*, Wiley, New York, 1973.

[22] Butcher, J. C., *Numerical Analysis of Ordinary Differential Equations*, Wiley, New York, 1987.

[23] Butcher, J. C., and Wanner, G., "Runge–Kutta Methods: Some Historical Notes," *Applied Numerical Mathematics*, Vol. 22, 1996, pp. 113–151.

[24] Yoh, J. J., and Zhong, X., "New Hybrid Runge–Kutta Methods for Unsteady Reactive Flow Simulation: Applications," *AIAA Journal*, Vol. 42, No. 8, 2004, pp. 1601–1611.

C. Kaplan
Associate Editor