SIAM J. NUMER. ANAL.
Vol. 38, No. 4, pp. 1369–1388

INEXACT SIMPLIFIED NEWTON ITERATIONS FOR IMPLICIT RUNGE-KUTTA METHODS[*]

LAURENT O. JAY[†]

Abstract. We consider possibly stiff and implicit systems of ordinary differential equations (ODEs). The major difficulty and computational bottleneck in the implementation of fully implicit Runge–Kutta (IRK) methods resides in the numerical solution of the resulting systems of nonlinear equations. To solve those systems we show that the use of inexact simplified Newton methods is efficient. Linear systems of the simplified Newton method are solved approximately with a preconditioned linear iterative method. Sufficient conditions ensuring local convergence of the inexact simplified Newton method for general nonlinear equations are given. The preconditioner that we use is based on the W-transformation of the RK coefficients and on the block-LU decomposition of the simplified Jacobian after W-transformation. A new code based on those techniques, SPARK3, is shown to be effective on two problems; the first one is a linear convection-diffusion problem and the second one a reaction-diffusion problem.

Key words. contraction, GMRES, implicit Runge–Kutta methods, inexact simplified Newton methods, linear iterative methods, nonlinear equations, ordinary differential equations, preconditioning, Richardson iterations, stiffness, W-transformation

AMS subject classifications. 34A65, 65F10, 65H10, 65L05, 65L06

PII. S0036142999360573

1. Introduction.

In this article we consider the application of fully implicit Runge–Kutta (IRK) methods to possibly stiff and implicit systems of ordinary differential equations (ODEs). The main difficulty and computational bottleneck in the implementation of IRK methods, such as those based on Gauss, Radau, or Lobatto points [24], is generally in the numerical solution of the resulting systems of nonlinear equations. In order to solve these systems efficiently we suggest the use of inexact simplified Newton methods, more precisely of simplified Newton-iterative methods. Linear systems of the simplified Newton method are solved approximately with a preconditioned linear iterative method, such as preconditioned versions of Richardson or GMRES iterations. We give sufficient conditions ensuring local convergence of the inexact simplified Newton method for general nonlinear equations. The preconditioner that we use is based on the W-transformation of the RK coefficients and on the block-LU decomposition of the simplified Jacobian after W-transformation. For an s-stage IRK method this requires the decomposition of s or $s-1$ independent submatrices of the same dimension as the differential system.

In section 2 the class of implicit systems of ODEs considered in this article is presented. In section 3 we define the application of IRK methods and describe briefly the W-transformation. In section 4 we consider and analyze inexact simplified Newton methods in a general context. In section 5 we motivate the use of inexact simplified Newton iterations for solving the systems of nonlinear equations of IRK methods. In section 6 we detail the approximate inverse matrix used as a preconditioner for the iterative solution of the linear systems of the simplified Newton method. This preconditioner is based on the W-transformation of the RK coefficients. In section 7

[*] Received by the editors August 24, 1999; accepted for publication (in revised form) June 21, 2000; published electronically October 31, 2000.
http://www.siam.org/journals/sinum/38-4/36057.html

[†] Department of Mathematics, 14 MacLean Hall, University of Iowa, Iowa City, IA 52242-1419 (ljay@math.uiowa.edu, na.ljay@na-net.ornl.gov).

we present two preconditioned linear iterative methods used for the solution to these linear systems; they are based on Richardson and GMRES iterations. A new code based on the aforementioned techniques, SPARK3, is presented in section 8. In section 9, we show that the code SPARK3 is effective on two problems; the first one is a linear convection-diffusion problem and the second one a reaction-diffusion problem.

2. The implicit system of ODEs. We consider a possibly stiff and implicit n-dimensional system of ODEs with a prescribed initial value

$$\frac{d}{dt}a(t,y) = f(t,y), \qquad y(t_0) = y_0, \tag{1}$$

where $y = (y^1, \ldots, y^n)^T \in \mathbf{R}^n$. We assume that

$$a_y(t,y) := \frac{\partial}{\partial y}a(t,y) \quad \text{is invertible} \tag{2}$$

in a neighborhood of the solution. By differentiating the left-hand side of (1) we get

$$a_y(t,y)\frac{d}{dt}y = f(t,y) - a_t(t,y), \tag{3}$$

where $a_t(t,y) := \frac{\partial}{\partial t}a(t,y)$. Hence, the assumption (2) implies that the above implicit system of ODEs (1) can be expressed as an explicit system of ODEs

$$\frac{d}{dt}y = a_y(t,y)^{-1}\left(f(t,y) - a_t(t,y)\right). \tag{4}$$

From a mathematical point of view the above formulations are equivalent. However, from a computational point of view they are not. The formulations (3) and (4) involve the terms $a_t(t,y)$ and $a_y(t,y)$, and (4) also involves the inverse $a_y(t,y)^{-1}$. In this article we will consider IRK methods directly applied to (1). The expressions $a_y(t,y)$, $a_t(t,y)$, and $a_y(t,y)^{-1}$ are thus not needed. When $a(t,y) \equiv y$ we obtain the usual system of ODEs $\frac{d}{dt}y = f(t,y)$. It is assumed that the implicit system of ODEs (1) presents some stiffness, so that it behooves us to consider the application of implicit methods, such as IRK methods.

3. IRK methods and the W-transformation. The application of IRK methods to the implicit system of ODEs (1) is as follows.

Definition 3.1. *One step of an s-*stage IRK method *applied to* (1) *with initial values y_0 at t_0 and stepsize h reads*

$$a(t_0 + c_i h, Y_i) - a(t_0, y_0) - h\sum_{j=1}^{s} a_{ij} f(t_0 + c_j h, Y_j) = 0 \quad \text{for } i = 1, \ldots, s, \tag{5a}$$

$$a(t_0 + h, y_1) - a(t_0, y_0) - h\sum_{i=1}^{s} b_i f(t_0 + c_i h, Y_i) = 0. \tag{5b}$$

The RK coefficients are given by the *weight vector* $b = (b_1, \ldots, b_s)^T$, the *node vector* $c = (c_1, \ldots, c_s)^T$, and the *RK coefficient matrix* $A = (a_{ij})_{i,j=1,\ldots,s}$. The equations (5a) define a nonlinear system of dimension $s \cdot n$ to be solved for the s *internal stages* Y_i for $i = 1, \ldots, s$. The numerical approximation at $t_0 + h$ is then given by the solution y_1 of the n-dimensional implicit system (5b). When the IRK method is *stiffly* *accurate*, i.e., when $a_{si} = b_i$ for $i = 1, \ldots, s$, there is no need to solve the nonlinear system for y_1 since this value is directly given by $y_1 = Y_s$. It will be assumed hereafter that the number s of stages satisfies $s \geq 2$.

One way to construct IRK methods is through the W-transformation of the RK coefficient matrices pioneered by Hairer and Wanner [22, 23]. Up until now this W-transformation has been used mostly for this aim and for other theoretical purposes such as the stability analysis of IRK methods, but to our knowledge it has never been used for any practical purpose. This article is an extension of a preliminary article by Jay and Braconnier [29] which introduced the W-transformation as a practical tool for the implementation of IRK methods. The W-transformation of the RK matrix A is defined by

$$X := W^T BAW, \tag{6}$$

where $B := \mathrm{diag}(b_1, \ldots, b_s)$ and the coefficients of the matrix W are given by $w_{ij} = P_{j-1}(c_i)$ where $P_k(x)$, the *kth shifted Legendre polynomial*, is given by

$$P_k(x) = \frac{\sqrt{2k+1}}{k!} \cdot \frac{d^k}{dx^k} \left(x^k (x-1)^k \right) = \sqrt{2k+1} \sum_{j=0}^{k} (-1)^{j+k} \binom{k}{j} \binom{j+k}{j} x^j.$$

For more details about the W-transformation we refer to [24, section IV.5] and [7]. Hereafter it will be assumed for the following matrices that

$X := W^T BAW$ is tridiagonal, $D := W^T BW$ is diagonal and regular,

two conditions which are satisfied for most IRK methods of interest, such as Gauss, Radau IA & IIA, Lobatto IIIA & IIIB & IIIC & IIIC* & IIID [7, 24, 27]. For these IRK methods the transformed matrix X and the matrix D are of the form

$$X = \begin{pmatrix} 1/2 & -\zeta_1 & & & O \\ \zeta_1 & 0 & \ddots & & \\ & \ddots & \ddots & -\zeta_{s-2} & \\ & & \zeta_{s-2} & 0 & \beta_{s-1,s} \\ O & & & \beta_{s,s-1} & \beta_{ss} \end{pmatrix}, \qquad D = \mathrm{diag}(1, 1, \ldots, 1, d_s), \tag{7}$$

where $\zeta_k = 1/\left(2\sqrt{4k^2-1}\right)$ and the missing coefficients $\beta_{s,s-1}, \beta_{s-1,s}, \beta_{ss}, d_s$ are given in Table 1. We also give in Table 1 the additional values α_s of Lemma 6.1. The forms (7) for the matrices X and D will be assumed hereafter.

4. Nonlinear equations and inexact simplified Newton iterations.

In this section we discuss inexact simplified Newton methods in a more general context than their application to the systems of nonlinear equations of IRK methods. We consider a system of equations

$$G(x) = 0, \tag{8}$$

where $G : \mathbf{R}^n \to \mathbf{R}^n$ is a nonlinear mapping satisfying the following assumptions:

A1. There exists an $x^* \in \mathbf{R}^n$ such that $G(x^*) = 0$;

A2. G is a continuously differentiable mapping in a neighborhood of x^*;

A3. $G'(x^*)$ is invertible.

TABLE 1
Values of certain coefficients for some IRK methods, $\sigma = \frac{2s-1}{s-1}$.

IRK method	$\beta_{s,s-1}$	$\beta_{s-1,s}$	β_{ss}	d_s	α_s
Gauss	ζ_{s-1}	$-\zeta_{s-1}$	0	1	$\frac{1}{2(2s-1)}$
Radau IA	ζ_{s-1}	$-\zeta_{s-1}$	$\frac{1}{4s-2}$	1	$\frac{1}{2s-1}$
Radau IIA	ζ_{s-1}	$-\zeta_{s-1}$	$\frac{1}{4s-2}$	1	$\frac{1}{2s-1}$
Lobatto IIIA	$\zeta_{s-1}\sigma$	0	0	σ	0
Lobatto IIIB	0	$-\zeta_{s-1}\sigma$	0	σ	0
Lobatto IIIC	$\zeta_{s-1}\sigma$	$-\zeta_{s-1}\sigma$	$\frac{\sigma}{2s-2}$	σ	$\frac{1}{s-1}$
Lobatto IIIC*	$\zeta_{s-1}\sigma$	$-\zeta_{s-1}\sigma$	$-\frac{\sigma}{2s-2}$	σ	0
Lobatto IIID	$\zeta_{s-1}\sigma$	$-\zeta_{s-1}\sigma$	0	σ	$\frac{1}{2s-2}$

4.1. The simplified Newton method.

A classical algorithm to solve a system of nonlinear equations satisfying these assumptions for a starting value x_0 sufficiently close to the locally unique zero x^* is given by the *simplified Newton method.* A sequence of iterates x_k is computed as follows:

ALGORITHM 4.1. *Simplified Newton method.*

```
0. Set $k := 0$;
1. While not convergence do
       Solve $G'(x_0)\Delta x_k = -G(x_k)$;
       Set $x_{k+1} := x_k + \Delta x_k$;
       Set $k := k + 1$;
   End While
2. $x := x_k$ is the approximate solution.
```

A major difficulty of these iterations is generally to solve the linear systems of equations with matrix $G'(x_0)$. In an *inexact simplified Newton method* these linear systems are solved only approximately, e.g., by a (preconditioned) linear iterative method and this can be called, using a standard terminology, a *simplified Newton-iterative method.* We emphasize the fact that by assuming the point x_0 to be sufficiently close to the solution x^*, we actually suppose that the linear models of the function $G(x)$ given by $G(x_k) + G'(x_0)(x - x_k)$ are good models. In particular we suppose that $\Phi(x) := x - G'(x_0)^{-1}G(x)$ is locally contractive. We can actually replace the matrix $G'(x_0)$ by an approximation A_0 provided this last property is satisfied, and the results given in this paper remain valid for such *inexact approximate (or modified) simplified Newton methods.* No globalization procedure can certainly be envisaged for inexact simplified Newton methods, in contrast to inexact Newton methods [2, 5, 11, 12, 30, 31], since we keep the same matrix $G'(x_0)$ during all iterations. Here we are really interested only in solving nonlinear equations for a starting value x_0 sufficiently close to the solution x^*. This is justified, for example, for the solution of the nonlinear equations of IRK methods (5) where the initial guess can be supposed to be close enough to the solution, and if not, the system of nonlinear equations can be modified by changing the stepsize h.

4.2. Simplified Newton iterations as fixed-point iterations.

The simplified Newton method can be interpreted equivalently as a fixed-point iteration process

$$x_{k+1} := \Phi(x_k) \quad \text{where} \quad \Phi(x) := x - G'(x_0)^{-1}G(x). \tag{9}$$

From

$$\Phi'(x) = I_n - G'(x_0)^{-1}G'(x),$$

we have $\Phi'(x_0) = 0$. Hence, if x_0 is sufficiently close to x^*, by continuity of Φ' there exists for any given norm $\|\cdot\|$ a value θ with $0 \leq \theta < 1$ such that $\Phi'(x)$ satisfies $\|\Phi'(x)\| \leq \theta$ for the corresponding induced matrix norm in a neighborhood of x^* containing x_0. Therefore, a direct consequence of the Taylor–Lagrange remainder formula or simply of the integral form of the mean value theorem (the Newton–Leibniz formula) is that $\Phi(x)$ is locally contractive. Hence, the sequence of iterates x_k in (9) satisfies

$$\|x_{k+1} - x^*\| = \|\Phi(x_k) - \Phi(x^*)\| \leq \theta\|x_k - x^*\| \tag{10}$$

and thus converges to the fixed-point x^* of $\Phi(x)$, i.e., $x^* = \Phi(x^*)$. This fixed-point x^* is also trivially a zero of $x - \Phi(x) = G'(x_0)^{-1}G(x)$, therefore of $G(x)$. The sequence of iterates x_k also satisfies

$$\|x_{k+1} - x_k\| = \|\Phi(x_k) - \Phi(x_{k-1})\| \leq \theta\|x_k - x_{k-1}\|. \tag{11}$$

Hence, if the linear systems of equations of the simplified Newton method are solved exactly, then the linear convergence factor θ can be estimated for $k \geq 1$ by

$$\theta_k := \frac{\|\Delta x_k\|}{\|\Delta x_{k-1}\|},$$

or more reliably by $\widehat{\theta}_1 := \theta_1$ and $\widehat{\theta}_k := \sqrt{\widehat{\theta}_{k-1}\theta_k}$ for $k \geq 2$. From the inequality

$$\begin{aligned} \|x_{k+1} - x^*\| &= \|x_{k+1} - x_{k+2} + x_{k+2} - x_{k+3} + x_{k+3} - \ldots - x^*\| \\ &\leq \left(\theta + \theta^2 + \theta^3 + \ldots\right)\|x_k - x_{k+1}\| = \frac{\theta}{1-\theta}\|\Delta x_k\| \end{aligned} \tag{12}$$

a natural stopping criterion for convergence of the simplified Newton iterations often used in practice is given by

$$\eta_k\|\Delta x_k\| \leq \kappa_1 \cdot TOL \quad \text{where} \quad \eta_k := \frac{\widehat{\theta}_k}{1-\widehat{\theta}_k},$$

TOL is an error tolerance, and κ_1 is a security factor such as $\kappa_1 = 0.03$.

4.3. Inexact fixed-point iterations. When the linear system of equations at each simplified Newton step is solved only approximately, we have what we call an *inexact simplified Newton method*. To motivate the use of such methods we can make the following observation: from the inequality (10), and given x_k and θ, there is generally no need to solve for x_{k+1} too accurately, i.e., with an accuracy much smaller than $\theta\|x_k - x^*\|$. This motivates a stopping criterion for the solution Δx_k of the linear system $G'(x_0)\Delta x_k = -G(x_k)$ based on θ and $\|\Delta x_{k-1}\|$. An accuracy of $\kappa_2\theta\|x_k - x^*\|$ for x_{k+1}, hence for Δx_k, should be acceptable where κ_2 is another security factor, such as $\kappa_2 = 0.1$. Thus, from (12) and (11), either of the two quantities

$$\kappa_2\theta^2\|\Delta x_{k-1}\|, \quad \kappa_2\frac{\theta^2}{1-\theta}\|\Delta x_{k-1}\|,$$

is a reasonable accuracy to determine x_{k+1}, thus Δx_k. In practice we can use either of the two approximations

$$\kappa_2 \widehat{\theta}_{k-1}^2 \|\Delta x_{k-1}\|, \quad \kappa_2 \frac{\widehat{\theta}_{k-1}^2}{1-\widehat{\theta}_{k-1}} \|\Delta x_{k-1}\|.$$

Instead of the exact sequence of iterates x_k we actually get another sequence of iterates $\tilde{x}_k$ with residual errors $\tilde{r}_{k+1} := G(\tilde{x}_k) + G'(x_0)\Delta\tilde{x}_k$ (be wary that the usual definition of a residual error in linear algebra is minus this quantity, but for the sake of simplicity we prefer the previous choice), where $\Delta\tilde{x}_k = \tilde{x}_{k+1} - \tilde{x}_k$ is the corresponding sequence of increments. This in turn leads to another sequence of estimates for the linear convergence factor

$$\tilde{\theta}_k := \frac{\|\Delta\tilde{x}_k\|}{\|\Delta\tilde{x}_{k-1}\|},$$

or more reliably $\bar{\theta}_1 := \tilde{\theta}_1$ and $\bar{\theta}_k := \sqrt{\bar{\theta}_{k-1}\tilde{\theta}_k}$ for $k \geq 2$.

We are interested in finding sufficient conditions, in terms of computable quantities, on the precision to which the linear systems of equations of the simplified Newton method should be solved, so that convergence is ensured and so that the linear convergence factor hardly deteriorates. In a more general context we are led to consider what we call *inexact fixed-point iterations*. Let θ be the contractivity factor of a contraction Φ in a given norm $\|\cdot\|$, i.e., $\|\Phi(x) - \Phi(y)\| \leq \theta\|x-y\|$ for $0 \leq \theta < 1$. Let x_0 be given and consider the fixed-point iterations $x_{k+1} := \Phi(x_k)$ for $k = 0, 1, 2, \ldots$. Let $\tilde{x}_0 := x_0$ and consider an approximate sequence $\tilde{x}_k$ to x_k satisfying $\tilde{x}_{k+1} = \Phi(\tilde{x}_k) + \delta\tilde{x}_{k+1}$, where $\delta\tilde{x}_{k+1}$ is the *direct error*. We call such a sequence an *inexact fixed-point sequence*. We can prove the following statement.

THEOREM 4.1. *Consider a contraction Φ for a given norm $\|\cdot\|$ with contractivity factor $0 \leq \theta < 1$ and an inexact fixed-point sequence $\tilde{x}_k$ to x_k. In addition assume that the increments $\Delta\tilde{x}_k := \tilde{x}_{k+1} - \tilde{x}_k$ satisfy*

$$\|\Delta\tilde{x}_{k+1}\| \leq \tilde{\theta}\|\Delta\tilde{x}_k\|, \quad k = 0, 1, 2, \ldots,$$

for a certain $\tilde{\theta}$ satisfying $0 \leq \tilde{\theta} < 1$. Then if the direct errors $\delta\tilde{x}_{k+1} := \tilde{x}_{k+1} - \Phi(\tilde{x}_k)$ satisfy

$$(13) \quad \|\delta\tilde{x}_1\| \leq \alpha_0\tilde{\theta}h(\tilde{\theta})\|\Delta\tilde{x}_0\|, \qquad \|\delta\tilde{x}_{k+1}\| \leq \alpha_k\tilde{\theta}^2 h(\tilde{\theta})\|\Delta\tilde{x}_{k-1}\|, \quad k = 1, 2, \ldots,$$

for a certain function $h(\tilde{\theta})$ and coefficients α_k, we have

$$(14) \qquad \|\tilde{x}_k - x_k\| \leq C_k\tilde{\theta}^k h(\tilde{\theta})\|\Delta\tilde{x}_0\|, \quad k = 0, 1, 2, \ldots,$$

where $C_k := \sum_{j=0}^{k-1} \alpha_{k-1-j}\kappa^j$ with $\kappa := \theta/\tilde{\theta}$.

Proof. For $k = 0$ we have $\|\tilde{x}_0 - x_0\| = 0$ and $C_0 = 0$. For $k = 1$ the result directly follows from the assumption (13) with $C_1 = \alpha_0$. The proof now can be made by induction on k. Assume the result to be true up to index k. For $k+1 \geq 2$ we have

$$\begin{aligned}\|\tilde{x}_{k+1} - x_{k+1}\| &\leq \|\delta\tilde{x}_{k+1}\| + \|\Phi(\tilde{x}_k) - \Phi(x_k)\| \leq \|\delta\tilde{x}_{k+1}\| + \theta\|\tilde{x}_k - x_k\| \\ &\leq \alpha_k\tilde{\theta}^2 h(\tilde{\theta})\|\Delta\tilde{x}_{k-1}\| + \theta C_k\tilde{\theta}^k h(\tilde{\theta})\|\Delta\tilde{x}_0\| \\ &\leq (\alpha_k + \kappa C_k)\tilde{\theta}^{k+1} h(\tilde{\theta})\|\Delta\tilde{x}_0\|.\end{aligned}$$

Hence $C_{k+1} = \alpha_k + \kappa C_k$. □

REMARK 4.2. *Note that we could replace the conditions* (13) *for $k \geq 1$ by the weaker conditions $\|\delta\tilde{x}_{k+1}\| \leq \alpha_k \tilde{\theta}^{k+1} h(\tilde{\theta}) \|\Delta\tilde{x}_0\|$. However, Theorem* 4.1 *emphasizes the fact that it is certainly safer and more reliable in practice to use* (13) *instead.*

The choice of the forcing terms $\alpha_0 \tilde{\theta} h(\tilde{\theta}) \|\Delta\tilde{x}_0\|$ and $\alpha_k \tilde{\theta}^2 h(\tilde{\theta}) \|\Delta\tilde{x}_{k-1}\|$ in (13) controls the convergence of the inexact fixed-point sequence similar to the forcing terms of inexact Newton methods [13]. However, the main difference here is that we assume the initial point x_0 to be already sufficiently close to the fixed-point x^*. Hence, we are concerned only with local convergence. The values of the coefficients α_k in (13) play an important role. We want to ensure that $C_k \tilde{\theta}^k \to 0$ when $k \to \infty$. This condition is satisfied, for example, by taking $\alpha_k := C\mu^k$ for two constants C and μ satisfying $C \geq 0$ and $0 \leq \mu < 1$. We obtain $C_k \tilde{\theta}^k = C((\mu\tilde{\theta})^k - (\mu\theta)^k)/(\mu - \kappa)$ if $\kappa \neq \mu$ or $C_k \tilde{\theta}^k = Ck\mu^{k-1}\tilde{\theta}^k$ if $\kappa = \mu$. Therefore $C_k \tilde{\theta}^k \to 0$ when $k \to \infty$.

From the inequality $\|\tilde{x}_k - x^*\| \leq \|\tilde{x}_k - x_k\| + \|x_k - x^*\|$ we get from (12) and (14) under the assumptions of Theorem 4.1

$$\|\tilde{x}_k - x^*\| \leq C_k \tilde{\theta}^k h(\tilde{\theta}) \|\Delta\tilde{x}_0\| + \frac{\theta}{1-\theta} \|\Delta x_{k-1}\| \leq C_k \tilde{\theta}^k h(\tilde{\theta}) \|\Delta\tilde{x}_0\| + \frac{\theta^k}{1-\theta} \|\Delta x_0\|.$$

Though not essential, it is natural to assume $\theta \leq \tilde{\theta}$ since otherwise this would mean that the inexact fixed-point sequence converges faster than its exact counterpart (provided the inexact fixed-point sequence converges to the locally unique fixed-point x^* of Φ). Defining the constant $c_0 := \|\Delta x_0\| / \|\Delta\tilde{x}_0\|$ to be discussed below, we obtain

$$\|\tilde{x}_k - x^*\| \leq \left(C_k h(\tilde{\theta}) + c_0 \frac{1}{1-\tilde{\theta}} \right) \tilde{\theta}^k \|\Delta\tilde{x}_0\|. \tag{15}$$

The coefficients α_k, the function $h(\tilde{\theta})$, the constant c_0, and the convergence factor $\tilde{\theta}$ all influence the convergence speed of the inexact fixed-point iterations. It is natural to choose the coefficients α_k and the function $h(\tilde{\theta})$ such that the two terms in brackets in (15) are approximately of the same size to avoid undersolving and oversolving. We can take, for example, $h(\tilde{\theta}) := 1/(1-\tilde{\theta})$ and determine α_k such that $C_k \approx c_0$. A safer choice for $h(\tilde{\theta})$, especially when $\tilde{\theta}$ is greatly overestimated, is given by $h(\tilde{\theta}) := 1$ since $1 \leq 1/(1-\tilde{\theta})$ for all $0 \leq \tilde{\theta} < 1$. By taking $\alpha_k = C\mu^k$, if $\delta\tilde{x}_1$ were available, then we could determine the smallest valid value of $\alpha_0 = C$ directly from (13) assuming that $\tilde{\theta}$ is known. Unfortunately, the direct error $\delta\tilde{x}_1 = \Delta\tilde{x}_0 - \Delta x_0$ cannot even be considered to be available. By obtaining $\tilde{x}_1 = x_0 + \Delta\tilde{x}_0 = \Phi(x_0) + \delta\tilde{x}_1$ sufficiently accurately, meaning $\|\delta\tilde{x}_1\| \ll \tilde{\theta} h(\tilde{\theta}) \|\Delta\tilde{x}_0\|$, any sufficiently positive value of $\alpha_0 = C$ will satisfy (13). The choice of the coefficients α_k, of the function $h(\tilde{\theta})$, and of the constant c_0 should also be dictated by the ratio C_{outer}/C_{inner} of the computational cost C_{outer} of one outer iteration (basically one function evaluation of G) over the cost C_{inner} of one inner iteration (basically approximately one iteration of the preconditioned linear iterative solver; note that this cost usually increases with the number of inner iterations). If this ratio is high then we should limit the number of outer iterations as much as possible by having C_k (and therefore α_k), $h(\tilde{\theta})$, c_0, and $\tilde{\theta}$ as small as possible. In reverse, if this ratio is small, then we should limit the number of inner iterations as much as possible; but we still should not increase the number of outer iterations significantly, which would in turn increase the total number of inner iterations. In both situations we should ensure $\tilde{\theta} \approx \theta$, and this may not hold if $\Delta\tilde{x}_0$ is not sufficiently close to Δx_0, i.e., we should have $c_0 = O(1)$. Both conditions $\|\delta\tilde{x}_1\| \ll \tilde{\theta} h(\tilde{\theta}) \|\Delta\tilde{x}_0\|$

and $\tilde{\theta} \approx \theta$ show the importance of obtaining the first inexact fixed-point iteration step $\tilde{x}_1 = x_0 + \Delta\tilde{x}_0$ sufficiently accurately.

The coefficients $\alpha_k = C\mu^k$ influence the precision to which the inexact fixed-point iterates $\tilde{x}_k$ are obtained. The main inconvenience of too small values for α_k is to obtain $\tilde{x}_k$ too accurately, leading to some loss of efficiency due to oversolving. Being too accurate, however, is certainly a safer strategy than the opposite! In reverse, being too inaccurate may lead to divergence of the inexact fixed-point iterations or convergence to an undesired value—two situations that we surely want to avoid. A value of $C = \alpha_0 = 1/3$ seems reasonable since as mentioned above we must obtain an accurate value for $\tilde{x}_1 = x_0 + \Delta\tilde{x}_0$. A practical and reasonable value for μ is given by $\mu = 2/3$. This implies that $C_k \le 3C$ for all $k \ge 0$ if $\kappa = \nu/\tilde{\nu} \le 1$.

4.4. A posteriori contraction test. The direct error $\delta\tilde{x}_2$ also cannot be considered to be available, and in the absence of an estimate for $\tilde{\theta}$, we must also obtain $\tilde{x}_2 = \tilde{x}_1 + \Delta\tilde{x}_1$ sufficiently accurately to ensure that (13) for $k = 1$ is also satisfied. One possibility is to shoot for a desired value of $\tilde{\theta}$ by having $\tilde{\theta}_{\text{desired}}$ in (13) for $k = 1$. If $\tilde{\theta}_{\text{desired}}$ is smaller than the actual value $\tilde{\theta}$, then we are just simply obtaining $\tilde{x}_2 = \tilde{x}_1 + \Delta\tilde{x}_1$ too accurately; but once again it is a safe strategy. Taking $\tilde{\theta}_{\text{desired}}$ too large may hinder the potential of a more rapid convergence which might have been obtained by taking $\tilde{\theta}_{\text{desired}}$ smaller. In any case, the value of $\tilde{\theta}_{\text{desired}}$ should be carefully selected by the user and should be as small as possible to ensure that the quantity $\alpha_1\tilde{\theta}^2_{\text{desired}}h(\tilde{\theta}_{\text{desired}})\|\Delta\tilde{x}_0\|$ in (13) is appropriate and such that a convergence factor of $\tilde{\theta}_{\text{desired}}$ would be really desired given the error tolerance that is aimed at and a possible maximum number of outer iterations that the user is ready to take.

Convergence of a sequence $\tilde{x}_k$ to a zero of a function H can also be shown and checked in practice under the a posteriori conditions below. In the context of inexact fixed-point iterations discussed above we can consider, for example, $H(x) := x - \Phi(x)$ where Φ is a contraction.

THEOREM 4.3. *Let $K \subset \mathbf{R}^n$ be a compact set and $H : K \to \mathbf{R}^n$ be a continuous mapping. Consider a sequence $\tilde{x}_k$ in K satisfying*

$$\|\tilde{x}_{k+1} - \tilde{x}_k\| \le \tilde{\theta}_k\|\tilde{x}_k - \tilde{x}_{k-1}\| \quad \text{for} \quad \tilde{\theta}_k \le \tilde{\theta} < 1, \tag{16a}$$

$$\|H(\tilde{x}_{k+1})\| \le \tilde{\rho}_k\|H(\tilde{x}_k)\| \quad \text{for} \quad \tilde{\rho}_k \le \tilde{\rho} < 1. \tag{16b}$$

Then the sequence $\tilde{x}_k$ converges in K to a zero x^* of H.

Proof. By the triangle inequality and from (16a), the sequence $\tilde{x}_k$ is a Cauchy sequence

$$\begin{aligned}\|\tilde{x}_{k+p} - \tilde{x}_k\| &= \left\|\sum_{i=0}^{p-1}\tilde{x}_{k+p-i} - \tilde{x}_{k+p-i-1}\right\| \le \sum_{i=0}^{p-1}\|\tilde{x}_{k+p-i} - \tilde{x}_{k+p-i-1}\| \\ &\le \sum_{i=0}^{p-1}\tilde{\theta}^{p-i-1}\|\tilde{x}_{k+1} - \tilde{x}_k\| \le \frac{1-\tilde{\theta}^p}{1-\tilde{\theta}}\|\tilde{x}_{k+1} - \tilde{x}_k\| \le \frac{\tilde{\theta}^k}{1-\tilde{\theta}}\|\tilde{x}_1 - \tilde{x}_0\|.\end{aligned}$$

Since this Cauchy sequence stays in the set K which is compact, hence complete, it must converge to a certain $x^* \in K$. From

$$\|H(\tilde{x}_k)\| \le \tilde{\rho}^k\|H(\tilde{x}_0)\|$$

with $\tilde{\rho} < 1$, we get $\lim_{k\to\infty} H(\tilde{x}_k) = 0$. By continuity of H we have $0 = \lim_{k\to\infty} H(\tilde{x}_k) = H(\lim_{k\to\infty}\tilde{x}_k) = H(x^*)$. □

We have considered in Theorem 4.3 a mapping H instead of the original mapping G in (8). When solving for a zero of G we can apply Theorem 4.3 to, for example, $H(x) := MG(x)$ where M is a regular matrix, e.g., $M = G'(x_0)^{-1}$ or $M = P^{-1}$ where P is a preconditioner of $G'(x_0)$. In the former case we have $H(x) = G'(x_0)^{-1}G(x) = x - \Phi(x)$ with Φ given by (9) and $H'(x_0) = I_n$. If the linear systems $G'(x_0)\Delta\tilde{x}_k = -G(\tilde{x}_k)$ are solved accurately enough without being too accurate (see Theorem 4.1), we have $\tilde{x}_{k+1} - \tilde{x}_k \approx -G'(x_0)^{-1}G(\tilde{x}_k) = -H(\tilde{x}_k)$. Hence, in this situation we may not need to check the second condition (16b) of Theorem 4.3, since when equality holds this condition is equivalent to (16a) with $\tilde{\rho}_k = \tilde{\theta}_k$ and $\tilde{\rho} = \tilde{\theta}$. We may check the condition (16b) using a preconditioner P of $G'(x_0)$ in $H(x) := P^{-1}G(x)$, but the absence of such a convergence test is still justified provided the preconditioner P is a good approximation to $G'(x_0)$, in the sense that $P^{-1}G'(x_0) = O(1)$; see [4] and the discussion below in subsection 4.5.

Theorem 4.3 resembles the contraction mapping theorem. A function H satisfying the above conditions is given, for example, by a contraction with fixed-point at $x^* = 0$. If $H(x) = x - \Phi(x)$ where Φ is a contraction then the zero x^* of H is also a fixed-point of Φ and vice versa. Therefore x^* is locally unique. If $H(x)$ is continuously differentiable in a neighborhood of x^* and if $H'(x^*)$ is nonsingular, then the zero x^* is also locally unique. Note that in Theorem 4.3 we have made no assumption on the differentiability of H and on how the sequence $\tilde{x}_k$ is generated. The first condition (16a) ensures that the iterates $\tilde{x}_k$ converge. The second condition (16b) ensures that these iterates converge to a zero of $H(x)$. Hence, we have simply separated a convergence condition for the sequence $\tilde{x}_k$ from a condition of sufficient decrease for $\|H(\tilde{x}_k)\|$. Any other convergence condition for the sequence $\tilde{x}_k$ and any other sufficient decrease condition for $\|H(\tilde{x}_k)\|$ can also ensure convergence to a zero of H. Therefore, in a certain sense Theorem 4.3 is trivial. Nevertheless, it seems justified to state and discuss it in details here since it has important practical implications and relations to other methods when solving systems of nonlinear equations. The condition of sufficient decrease (16b) resembles the inexact Newton condition [11, 30, 31]

$$(17) \qquad \|H(\tilde{x}_k) + H'(\tilde{x}_k)(\tilde{x}_{k+1} - \tilde{x}_k)\| \leq \tilde{\rho}_k \|H(\tilde{x}_k)\| \quad \text{for} \quad \tilde{\rho}_k \leq \tilde{\rho} < 1$$

which is nothing else but (16b) with $H(\tilde{x}_{k+1})$ replaced by its first-order Taylor series at $\tilde{x}_k$. The close relationship between these two conditions is another justification of the inexact Newton condition (17). However, the sufficient decrease condition (16b) is stronger than (17) and is also more natural in the context of simplified Newton iterations. It avoids any computation involving $H'(\tilde{x}_k)$, but it requires evaluations of the function H, whereas (17) can usually be checked directly in the inner iterations of a linear iterative method at almost no cost. For simplified Newton iterations, modifying the inexact Newton condition (17) by replacing $H'(\tilde{x}_k)$ with $H'(x_0)$ gives

$$(18) \qquad \|H(\tilde{x}_k) + H'(\tilde{x}_0)(\tilde{x}_{k+1} - \tilde{x}_k)\| \leq \tilde{\rho}_k \|H(\tilde{x}_k)\| \quad \text{for} \quad \tilde{\rho}_k \leq \tilde{\rho} < 1.$$

This is generally not sufficient to ensure convergence to a zero of H. Nevertheless, this motivates a condition for the (linear) residual errors to be satisfied, to be discussed in the following subsection.

4.5. Residual errors and convergence of inexact simplified Newton iterations. Now we concentrate our discussion on the situation where $H(x) := P^{-1}G(x)$ and the matrix P is a preconditioner of $G'(x_0)$. Here we do not make any assumption on the quality of the preconditioner P. We consider an inexact simplified Newton method applied to $G(x) = 0$. A preconditioned linear iterative solver applied to $G'(x_0)\Delta\tilde{x}_k = -G(\tilde{x}_k)$ to obtain $\tilde{x}_{k+1} = \tilde{x}_k + \Delta\tilde{x}_k$ usually does not monitor directly the error

$$\delta\tilde{x}_{k+1} = \Delta\tilde{x}_k + G'(x_0)^{-1}G(\tilde{x}_k) = G'(x_0)^{-1}(G'(x_0)\Delta\tilde{x}_k + G(\tilde{x}_k)) \tag{19}$$

as could be desired in order to apply Theorem 4.1, but it usually monitors the residual errors $\tilde{r}_{k+1}$ and the preconditioned residual errors

$$P^{-1}\tilde{r}_{k+1} = P^{-1}(G'(x_0)\Delta\tilde{x}_k + G(\tilde{x}_k)) = P^{-1}G'(x_0)\delta\tilde{x}_{k+1} \tag{20}$$

which differ from the above expression (19) by the matrix factor $P^{-1}G'(x_0)$. This implies some slight modifications to obtain a convergence result similar to Theorem 4.1. Note that convergence can also be checked a posteriori with the conditions (16) of Theorem 4.3.

Analogous to (13) and (18) we suggest controlling the preconditioned residual errors as follows:

$$\|P^{-1}\tilde{r}_{k+1}\| \le \alpha_k\tilde{\nu}h(\tilde{\nu})\|P^{-1}G(\tilde{x}_k)\| \quad k = 0, 1, 2, \ldots, \tag{21}$$

for a certain function $h(\tilde{\nu})$ and coefficients α_k. For $P = I$ we obtain the condition (18) with $H = G$ and $\tilde{\rho}_k = \alpha_k\tilde{\nu}h(\tilde{\nu})$. For $P \approx G'(x_0)$, by (20) this condition is almost equivalent to (13) with α_j replaced by $\alpha_j/(1-\alpha_j\tilde{\nu}h(\tilde{\nu}))$, $\|\Delta\tilde{x}_k\| \approx \|G(x_0)^{-1}G(\tilde{x}_k)\|$, and $\tilde{\theta} \approx \tilde{\nu}$. This can be seen by writing $P = G(x_0)(I - E)$ where the matrix E is a small perturbation matrix. From (20) we get $P^{-1}\tilde{r}_{k+1} = (I-E)^{-1}\delta\tilde{x}_{k+1}$ and from $-G'(x_0)^{-1}G(\tilde{x}_k) = \Delta\tilde{x}_k - \delta\tilde{x}_{k+1}$ we get $P^{-1}G(\tilde{x}_k) = -(I-E)^{-1}(\Delta\tilde{x}_k - \delta\tilde{x}_{k+1})$. Hence (21) can be expressed by

$$\|(I-E)^{-1}\delta\tilde{x}_{k+1}\| \le \alpha_k\tilde{\nu}h(\tilde{\nu})\|(I-E)^{-1}(\Delta\tilde{x}_k - \delta\tilde{x}_{k+1})\|.$$

As mentioned in the discussion after Theorem 4.1, it is important to obtain the first iterate $\tilde{x}_1 = \tilde{x}_0 + \Delta\tilde{x}_0$ sufficiently accurately, so that convergence of the outer iterations is not hindered. Now we can state the main result of this section.

THEOREM 4.4. *Consider $\Phi(x) := x - H(x)$ where $H(x) = G'(x_0)^{-1}G(x)$ and an inexact simplified Newton sequence $\tilde{x}_k$ for $G(x) = 0$ (also an inexact fixed-point sequence for $x = \Phi(x)$) starting at $\tilde{x}_0 = x_0$. We denote the exact sequence by $x_{k+1} := \Phi(x_k)$. Assume that the increments $\Delta\tilde{x}_k := \tilde{x}_{k+1} - \tilde{x}_k$ satisfy*

$$\|\Delta\tilde{x}_{k+1}\|_* \le \tilde{\nu}\|\Delta\tilde{x}_k\|_*, \quad k = 0, 1, 2, \ldots,$$

in the norm $\|v\|_* := \|P^{-1}G'(x_0)v\|$ for a certain $\tilde{\nu}$ satisfying $0 \le \tilde{\nu} < 1$, i.e.,

$$\|P^{-1}G'(x_0)\Delta\tilde{x}_{k+1}\| \le \tilde{\nu}\|P^{-1}G'(x_0)\Delta\tilde{x}_k\|, \quad k = 0, 1, 2, \ldots,$$

where P is a preconditioner to the matrix $G(x_0)$. Assume that Φ is locally contractive with contractivity factor $0 \le \nu < 1$ in the norm $\|\cdot\|_*$. Then if the preconditioned residual errors $P^{-1}\tilde{r}_{k+1} = P^{-1}(G'(x_0)\Delta\tilde{x}_k + G(\tilde{x}_k))$ satisfy (21) for a certain function h, and the coefficients α_k satisfy $\alpha_k\tilde{\nu}h(\tilde{\nu}) < 1$, we have

$$\|\tilde{x}_k - x_k\|_* \le D_k\tilde{\nu}^k h(\tilde{\nu})\|\Delta\tilde{x}_0\|_*, \quad k = 0, 1, 2, \ldots,$$

where $D_k := \sum_{j=0}^{k-1}\beta_{k-1-j}\kappa^j$ with $\kappa := \nu/\tilde{\nu}$ and $\beta_j := \alpha_j/(1-\alpha_j\tilde{\nu}h(\tilde{\nu}))$.

Proof. For $k = 0$ we have $\|\tilde{x}_0 - x_0\| = 0$ and $D_0 = 0$. The proof now can be made by induction on k. Assume the result to be true up to index k. For $k + 1 \geq 1$ we have

$$\|\tilde{x}_{k+1} - x_{k+1}\|_* \leq \|P^{-1}G'(x_0)\delta\tilde{x}_{k+1}\| + \|\Phi(\tilde{x}_k) - \Phi(x_k)\|_*.$$

As seen above we have $P^{-1}G'(x_0)\delta\tilde{x}_{k+1} = P^{-1}\tilde{r}_{k+1}$. Using the assumption (21) and rewriting $G(\tilde{x}_k) = G'(x_0)\delta\tilde{x}_{k+1} - G'(x_0)\Delta\tilde{x}_k$ we obtain

$$\|P^{-1}\tilde{r}_{k+1}\| \leq \alpha_k\tilde{\nu}h(\tilde{\nu})\left(\|P^{-1}\tilde{r}_{k+1}\| + \|\Delta\tilde{x}_k\|_*\right).$$

This implies that $\|P^{-1}\tilde{r}_{k+1}\| \leq \beta_k\tilde{\nu}h(\tilde{\nu})\|\Delta\tilde{x}_k\|_*$. Thus

$$\begin{aligned}\|\tilde{x}_{k+1} - x_{k+1}\|_* &\leq \beta_k\tilde{\nu}h(\tilde{\nu})\|\Delta\tilde{x}_k\|_* + \nu\|\tilde{x}_k - x_k\|_* \\ &\leq (\beta_k + \kappa D_k)\tilde{\nu}^{k+1}h(\tilde{\nu})\|\Delta\tilde{x}_0\|_*,\end{aligned}$$

giving the desired result with $D_{k+1} = \beta_k + \kappa D_k$. □

Remark 4.5.

1. *Note again that the choice of the forcing coefficients $\alpha_k\tilde{\nu}h(\tilde{\nu})$ in* (21) *controls the convergence of the inexact simplified Newton sequence similar to the forcing coefficients of inexact Newton methods* [13]. *However, we emphasize again that the main difference here is that we assume the initial point x_0 to be already sufficiently close to the exact solution x^*. Hence, we are concerned only with local convergence. To have $\beta_j = C\mu^j$ we must take $\alpha_j := C\mu^j/(1 + C\mu^j\tilde{\nu}h(\tilde{\nu}))$.*
2. *Note that to compute $\|\Delta\tilde{x}_k\|_*$ we need $P^{-1}G'(x_0)\Delta\tilde{x}_k$ which can be expressed as*

$$P^{-1}G'(x_0)\Delta\tilde{x}_k = P^{-1}\tilde{r}_{k+1} - P^{-1}G(\tilde{x}_k)$$

and the two quantities on the right-hand side are readily available (see (21)*).*

In the context of the solution of the nonlinear equations of IRK methods, we can take as the first estimate of $\tilde{\nu}$ for the computation at a given timestep of the first iterate $\tilde{x}_1$, the quantity $\tilde{\nu}_{\text{prev}}$ obtained at the previous timestep. From (21) above we should choose the precision to obtain $P^{-1}r_1$ to be at least $\alpha_0\tilde{\nu}_{\text{prev}}h(\tilde{\nu}_{\text{prev}})\|P^{-1}G(x_0)\|$. It is certainly safer to take an even lower value due to possible stepsize changes, etc., which could lead to a significant difference between $\tilde{\nu}_{\text{prev}}$ and $\tilde{\nu}$. In the code SPARK3 (see section 8) we take $\kappa_2\alpha_0\tilde{\nu}_{\text{prev}}^{1.5}\|P^{-1}G(x_0)\|$ where κ_2 is a security factor, e.g., $\kappa_2 = 0.1$, except for the first timestep where we solve for $\tilde{x}_1$ very accurately, e.g., up to the desired tolerance TOL, since no estimate of $\tilde{\nu}$ is generally available.

5. The nonlinear systems of IRK methods and simplified Newton iterations.

Various iteration schemes have been suggested to solve the system of nonlinear equations (5a) for the s internal stages Y_i [9, 10, 18, 19, 25, 34]. These methods can be viewed as *ad hoc* modifications to simplified Newton iterations. They do not usually iterate at the linear algebra level. They are generally tuned to the scalar linear Dahlquist's test equation $y' = \lambda y$ for $\mathrm{Re}(\lambda) \leq 0$. Unfortunately, none of them is asymptotically exact for stiff systems, not even the internal stages Y_i for this simple Dahlquist's test equation. In contrast the inexact simplified Newton technique presented in this article is by construction asymptotically correct.

In a standard approach the system of nonlinear equations (5a) for the *s internal stages* is solved by simplified Newton iterations with approximate Jacobian matrix

$$L := I_s \otimes M - hA \otimes J \quad \text{where} \quad M := a_y(t_0, y_0), \quad J := f_y(t_0, y_0). \tag{22}$$

The symbol $\otimes$ denotes the tensor product, and I_s is the identity matrix in $\mathbf{R}^s$. Simplified Newton iterations read

$$L\Delta Y^k = -F(Y^k), \qquad Y^{k+1} = Y^k + \Delta Y^k, \qquad k = 0, 1, 2, \ldots,$$

where $Y := (Y_1^T, \ldots, Y_s^T)^T$ is a vector collecting the s internal stages Y_i for $i = 1, \ldots, s$ and $F(Y)$ corresponds to the left-hand side of (5a). Hence, simplified Newton iterations require the solution of $(s \cdot n)$-dimensional linear systems with the above approximate Jacobian matrix L. The direct decomposition of this matrix L is generally inefficient when $s \geq 2$. By exploiting its special structure, its decomposition cost can be greatly improved. For example, by diagonalizing the RK coefficient matrix A

$$S^{-1}AS = \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_s)$$

the approximate Jacobian matrix can be transformed into a block-diagonal matrix

$$(23) \quad (S^{-1} \otimes I_n)L(S \otimes I_n) = I_s \otimes M - h\Lambda \otimes J = \begin{pmatrix} M - \lambda_1 hJ & & O \\ & \ddots & \\ O & & M - \lambda_s hJ \end{pmatrix}.$$

This transformation dramatically reduces the number of operations and allows for parallelism. Unfortunately, almost all eigenvalues of standard IRK methods arise as conjugate complex pairs. This significantly increases the decomposition cost of the transformed approximate Jacobian matrix (23) [24, section IV.8] and impairs parallelism. Moreover, if several distinct IRK methods are used in a partitioned and/or additive way, such as for SPARK methods [27], this diagonalization procedure cannot be applied since the different RK matrices generally possess distinct eigenvectors. Ideally, the decomposition cost of the Jacobian matrix for s-stage IRK methods should be equivalent to at most s independent decompositions of submatrices of dimension n.

In this article we present a different approach aimed at reducing the computational load. Instead of solving exactly the linear systems of the simplified Newton iterations, we solve approximately and iteratively a preconditioned version of those linear systems. The use of linear iterative methods for the solution of implicit integration methods was considered in [3, 8, 15], with an emphasis on preconditioning in [4]. Here we use a preconditioner requiring at most s independent decompositions of matrices of dimension n. Hence, the decomposition cost for a parallel implementation is equivalent to the cost for the implicit Euler method. A detailed presentation of the preconditioner is given in section 6.

6. Preconditioning the linear systems. Using the W-transformation (6) for the approximate Jacobian matrix L in (22), at each simplified Newton iteration we obtain a linear system

$$(24) \qquad Kx = b$$

with a block-tridiagonal matrix

$$(25) \qquad \begin{aligned} K &= (W^T B \otimes I_n)L(W \otimes I_n) \\ &= D \otimes M - hX \otimes J = \begin{pmatrix} E_1 & F_1 & & & O \\ G_1 & E_2 & F_2 & & \\ & \ddots & \ddots & \ddots & \\ & & G_{s-2} & E_{s-1} & F_{s-1} \\ O & & & G_{s-1} & E_s \end{pmatrix}, \end{aligned}$$

where the $n \times n$ blocks are given by

$$E_1 = M - \frac{1}{2}hJ, \quad E_i = M \quad \text{for } i = 2, \ldots, s-1, \quad E_s = d_s M - \beta_{ss} hJ, \tag{26a}$$

$$F_i = \zeta_i hJ \quad \text{for } i = 1, \ldots, s-2, \quad F_{s-1} = -\beta_{s-1,s} hJ, \tag{26b}$$

$$G_i = -\zeta_i hJ \quad \text{for } i = 1, \ldots, s-2, \quad G_{s-1} = -\beta_{s,s-1} hJ. \tag{26c}$$

A way of solving (24) could be to use the block-LU decomposition [16, 17] of (25)

$$K = \begin{pmatrix} I_n & & & & O \\ G_1 H_1^{-1} & I_n & & & \\ & \ddots & \ddots & & \\ & & G_{s-2}H_{s-2}^{-1} & I_n & \\ O & & & G_{s-1}H_{s-1}^{-1} & I_n \end{pmatrix} \begin{pmatrix} H_1 & F_1 & & & O \\ & H_2 & F_2 & & \\ & & \ddots & \ddots & \\ & & & H_{s-1} & F_{s-1} \\ O & & & & H_s \end{pmatrix},$$

where the blocks H_i are recursively given by

$$H_1 = E_1, \qquad H_i = E_i - G_{i-1} H_{i-1}^{-1} F_{i-1} \quad \text{for } i = 2, \ldots, s, \tag{27}$$

and are assumed to be regular. Subdividing the solution vector x, the right-hand side b of (24), and an intermediate vector y into s n-dimensional subvectors

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{s-1} \\ x_s \end{pmatrix}, \; b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{s-1} \\ b_s \end{pmatrix}, \; y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{s-1} \\ y_s \end{pmatrix}, \quad x_i, b_i, y_i \in \mathbf{R}^n \quad \text{for } i = 1, \ldots, s,$$

the linear system (24) can be solved using block forward and backward substitutions

$$\begin{aligned} &y_1 = b_1, \qquad y_i = b_i - G_{i-1} H_{i-1}^{-1} y_{i-1} \quad \text{for } i = 2, \ldots, s, \\ &x_s = H_s^{-1} y_s, \qquad x_i = H_i^{-1}(y_i - F_i x_{i+1}) \quad \text{for } i = s-1, \ldots, 1. \end{aligned}$$

From (26) and (27) the blocks H_i are given by

$$\begin{aligned} &H_1 = M - \frac{1}{2}hJ, \qquad H_i = M + \zeta_{i-1}^2 h^2 J H_{i-1}^{-1} J \quad \text{for } i = 2, \ldots, s-1, \\ &H_s = d_s M - \beta_{ss} hJ - \beta_{s,s-1}\beta_{s-1,s} h^2 J H_{s-1}^{-1} J. \end{aligned}$$

Since each block H_i for $i \geq 2$ depends on H_{i-1}^{-1}, the above recursion is not easily parallelizable. Moreover, we should also assume that all blocks H_i are regular, a condition which can actually be violated even if $M - hJ$ is assumed to be invertible for all $h \geq 0$. The computational load of such a procedure would be prohibitive anyway compared to the use of the diagonalization of the RK coefficient matrix in (23).

We now present our central idea. Instead of solving (24) directly, we apply a linear iterative method to the left-preconditioned linear system

$$P^{-1} K x = P^{-1} b. \tag{28}$$

We choose the preconditioner P to be given by the approximate block-LU decomposition of K based on independent approximations $\widetilde{H}_i$ of H_i. We set

$$P := \begin{pmatrix} I_n & & & & O \\ G_1\widetilde{H}_1^{-1} & I_n & & & \\ & \ddots & \ddots & & \\ & & G_{s-2}\widetilde{H}_{s-2}^{-1} & I_n & \\ O & & & G_{s-1}\widetilde{H}_{s-1}^{-1} & I_n \end{pmatrix} \begin{pmatrix} \widetilde{H}_1 & F_1 & & & O \\ & \widetilde{H}_2 & F_2 & & \\ & & \ddots & \ddots & \\ & & & \widetilde{H}_{s-1} & F_{s-1} \\ O & & & & \widetilde{H}_s \end{pmatrix} \tag{29}$$

with

$$\widetilde{H}_i := M - \gamma_i hJ \quad \text{for } i = 1, \ldots, s-1, \qquad \widetilde{H}_s := d_s\left(M - \frac{\gamma_s}{d_s}hJ\right), \tag{30}$$

where

$$\gamma_1 = \frac{1}{2}, \qquad \gamma_i = \frac{\zeta_{i-1}^2}{\gamma_{i-1}} \quad \text{for } i = 2, \ldots, s-1, \qquad \gamma_s = \beta_{ss} - \frac{\beta_{s,s-1}\beta_{s-1,s}}{\gamma_{s-1}}. \tag{31}$$

For $M = I_n$ this corresponds to the preconditioner derived in [29]. For a general regular matrix M we find the above preconditioner by simply premultiplying the linear system (24) onto the left by $(I_n \otimes M^{-1})$, applying the derivation of [29] to the matrix $(I_n \otimes M^{-1})K$, and then by multiplying the result onto the left by $(I_n \otimes M)$. Each $\widetilde{H}_i$ can be formed and decomposed independently, making these operations fully parallelizable. The coefficients γ_i have been chosen so that $\widetilde{H}_i^{-1}H_i \approx I_n$ when $(M - hJ)^{-1}(-hJ) \approx I_n$ for all $h \geq h_0 > 0$. For example, for $i = 2$ and if $s \geq 3$ we have

$$H_2 = M + \zeta_1^2 hJ\left(M - \frac{1}{2}hJ\right)^{-1} hJ.$$

We see that if M is negligible compared to hJ, then we can approximate H_2 by

$$\widetilde{H}_2 := M - 2\zeta_1^2 hJ$$

which is also correct when $hJ = 0$. In approximation theory this corresponds in a certain sense to approximating the polynomial $1 + \zeta_1^2 z(1 - z/2)^{-1}z$ by $1 - 2\zeta_1^2 z$. This process can be repeated for all matrices H_i leading to (30). If the RK coefficient matrix is invertible, i.e., if $\gamma_s \neq 0$, then the above preconditioner is asymptotically exact for stiff systems, like, for example, $y' = \lambda y$ when $|h\lambda| \to \infty$. We note that the preconditioner is consistent in the sense that for $h = 0$ we have $P = K = I_s \otimes M$. Moreover, its construction is valid for any choice of the Jacobians M and J of the differential system. Approximations to these Jacobians can also be used.

In the following result we give explicit formulas for the coefficients γ_i.

LEMMA 6.1. *The coefficients γ_i of* (31) *satisfy*

$$\gamma_i = \frac{1}{2(2i-1)} \text{ for } i = 1, \ldots, s-1.$$

For $i = s$ the coefficient $\alpha_s := \gamma_s/d_s$ of the Gauss, Radau IA & IIA, Lobatto IIIA & IIIB & IIIC & IIIC & IIID methods is as given in Table* 1.

Proof. The proof for $i = 1, \ldots, s-1$ can be done by induction. For $i = 1$ we have $\gamma_1 = 1/2$. Suppose now that the result is correct for a given index i. Since $\zeta_i^2 = 1/\big(4(4i^2-1)\big)$ we obtain

$$\gamma_{i+1} = \frac{\zeta_i^2}{\gamma_i} = \frac{2(2i-1)}{4(4i^2-1)} = \frac{1}{2(2i+1)}.$$

For γ_s in (31) we can use the equality $\zeta_{s-1}^2/\gamma_{s-1} = 1/(2(2s-1))$ which follows directly from above. From Table 1 we get the following results: for Gauss methods we have $\gamma_s = \zeta_{s-1}^2/\gamma_{s-1} = 1/(2(2s-1))$; for Radau IA & IIA we have $\gamma_s = 1/(4s-2) + \zeta_{s-1}^2/\gamma_{s-1} = 1/(2s-1)$; for Lobatto IIIA & IIIB we have $\gamma_s = 0$; for Lobatto IIIC we have $\gamma_s = \sigma\left(1/(2s-2) + \sigma\zeta_{s-1}^2/\gamma_{s-1}\right) = \sigma/(s-1)$; for Lobatto IIIC* we have $\gamma_s = \sigma(-1/(2s-2) + \sigma\zeta_{s-1}^2/\gamma_{s-1}) = 0$; for Lobatto IIID we have $\gamma_s = \sigma\zeta_{s-1}^2/\gamma_{s-1} = \sigma/(2s-2)$. □

For most standard IRK methods, a factor of two or more in the number of operations over the classical approach of diagonalizing the RK coefficient matrix [24] can be saved in terms of matrix decompositions by using the new preconditioning technique. Moreover, this new approach can be extended to SPARK methods [27, 29]. We see from Lemma 6.1 and from Table 1 that the coefficients γ_i are distinct from each other, but that $\alpha_s = \gamma_s/d_s$ may be equal to zero or to one of the coefficients γ_i for $i = 1, \ldots, s-1$. In this situation the decomposition of $\widetilde{H}_s$ is directly available. For $k = 1, 2, 3, \ldots$ the $s = (4k-1)$-stage Lobatto IIIC methods satisfy $\alpha_s = \gamma_k$ ($s = 3, 7, 11, \ldots$). For low values of s this leads to significant computational savings. It makes a method like the 3-stage Lobatto IIIC attractive especially for large-scale problems where direct methods are not applicable to solve the resulting linear systems. We also stress the facts that the stability properties of Lobatto IIIC methods are strong [24] and that Lobatto-type methods can be extended naturally to integrate differential-algebraic equations [26, 27].

7. Iterative solution of the linear systems.

The linear system (24) can be solved by a linear iterative method applied to the left-preconditioned system (28) using the preconditioner described in section 6. The nice feature of linear iterative methods is that it is not required to compute and to store the Jacobian matrices explicitly. Only matrix-vector products are needed.

Starting from $x_0 := 0$, the simplest linear iterative method is given by *preconditioned Richardson iterations* (PRIs)

$$x_{k+1} := (I - P^{-1}K)x_k + P^{-1}b \quad \text{for } k = 0, 1, 2, \ldots. \tag{32}$$

Richardson iterations are the simplest linear first-order iterations. If $\rho(I - P^{-1}K) < 1$, where ρ denotes the spectral radius of a matrix, then in exact arithmetic PRIs converge linearly, otherwise PRIs generally diverge [17]. Another possibility is to use linear iterative methods based on Krylov subspaces $\{r_0, P^{-1}Kr_0, \ldots, (P^{-1}K)^m r_0\}$ such as the GMRES method [33], where r_0 is usually the residual error $P^{-1}b - P^{-1}Kx_0$ of an initial approximation x_0. Basically, the GMRES algorithm builds an iterate x_m in these Krylov subspaces minimizing the 2-norm of the residual error $\|P^{-1}b - P^{-1}Ky\|_2$. Note that for the GMRES method, convergence is theoretically ensured after a finite number of iterations, but its convergence speed greatly depends on the spectral distribution of the preconditioned matrix $P^{-1}K$. The more the eigenvalues of $P^{-1}K$ are clustered and are close to a single point away from the origin, the better the convergence behavior [6, 20]. We will not give details about the GMRES algorithm here. The interested reader can consult, for example, [14, 17, 20, 32, 33]. The use of preconditioned linear iterative solvers for systems of differential equations was first considered in the context of implicit multistep methods and of differential-algebraic equations by Brown, Hindmarsh, and Petzold in [4].

8. A new code: SPARK3. We have developed a new code named SPARK3 for the numerical solution of (1). It is based on 3-stage IRK methods. The user can choose between the family of Lobatto IIIA & IIIB & IIIC & IIIC* & IIID coefficients, the Radau IIA coefficients, and the Gauss coefficients. The variable y is partitioned into $y = (u^T, v^T)^T$ and $a(t, y)$ of (1) is assumed to be of the form

$$a(t, y) = \begin{pmatrix} u \\ e(t, y) \end{pmatrix}.$$

We have incorporated in SPARK3 the choice between two linear iterative solvers. The first is a preconditioned version of Richardson iterations. The second makes use of preconditioned GMRES(m) iterations with a restart parameter m which can be set by the user. For the GMRES(m) iterations we have made slight modifications to the code `drive_dgmres` written for double precision arithmetic computation and developed at CERFACS [14]. This GMRES code is implemented using reverse communication and was therefore conveniently incorporated into SPARK3. The matrix multiplications and the dot products are all to be done outside the code `drive_dgmres`.

In SPARK3 we consider a scaled 2-norm, the *TOL-norm*, which depends on absolute and relative error tolerances for each component, $ATOL_i$ and $RTOL_i$, respectively, to be specified by the user

$$\|y\|_{TOL} := \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i}{D_i}\right)^2}, \qquad D_i := ATOL_i + RTOL_i|y_i|. \tag{33}$$

This is the natural norm to be considered in a code for differential equations [21]. To estimate the discretization error, the error of the simplified Newton method, and the error of the linear iterative solver, we use the TOL-norm with $|y_i|$ in D_i (33) replaced by $\max(|y_{0i}|, |y_{1i}|)$ where y_0, y_1 denote the numerical approximations at both extremities of the current interval of integration.

SPARK3 makes calls to BLAS routines and depending on the user choice also to LAPACK routines [1]. The preconditioning and matrix-vector products with the Jacobian of the differential system can be made internally or externally. The Jacobian matrix-vector products can be made as standard products using the computed Jacobian, can be Jacobian-free by using finite differences as in [4], or can be supplied by the user in any desired way. A user's guide to SPARK3 is in preparation [28]. The most current version of SPARK3 is available on the World Wide Web at http://www.math.uiowa.edu/~ljay/SPARK3.html.

9. Numerical results. For the first numerical experiment we consider a one-dimensional linear convection-diffusion equation

$$\frac{\partial u}{\partial t} = \alpha\frac{\partial^2 u}{\partial x^2} - \beta\frac{\partial u}{\partial x}, \tag{34}$$

where $\alpha \geq 0$ and β are both constant parameters. The initial condition at $t = 0$ is given by $u(0, x) = \sin(x)$. We consider periodic boundary conditions $u(t, x) =$

TABLE 2
Results of SPARK3 on the convection-diffusion equations (34).

Error tolerance TOL	10^{-3}	10^{-6}	10^{-9}	10^{-12}
TOL-norm error at t_{end}	0.59	0.51	0.19	0.58
CPU-time [s]	0.68	1.24	4.27	20.08
number of steps	9	14	46	233
number of rejected steps	0	0	0	0
number of function evaluations	33	45	141	702
number of inexact simplified Newton iterations	11	15	47	234
number of J evaluations	1	1	1	1
number of P decompositions	9	10	10	10
number of P solves	90	171	628	3025
number of linear iterations	65	130	489	2325
number of matrix-vector products	79	156	581	2791

$u(t, x + 2\pi)$; hence we can restrict x to $[0, 2\pi[$. The exact solution to this problem is given by $u(t, x) = e^{-\alpha t} \sin(x - \beta t)$ We apply the method of lines by discretizing the spatial operators using centered differences for the diffusion term and backward differences for the convection term (upwinding). We consider a grid of N points $x_i = i/(N+1)$ for $i = 1, \ldots, N, \Delta x = 1/(N+1)$. We obtain a large system of N stiff ODEs. The Jacobian J (22) is of the form

$$J = \frac{\alpha}{(\Delta x)^2} \begin{pmatrix} 2 & -1 & & & -1 \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ -1 & & & -1 & 2 \end{pmatrix} - \frac{\beta}{\Delta x} \begin{pmatrix} 1 & & & & -1 \\ -1 & 1 & & & \\ & \ddots & \ddots & & \\ & & -1 & 1 & \\ & & & -1 & 1 \end{pmatrix}.$$

We neglect the three off-band terms in the matrix J to obtain banded matrices $\widetilde{H}_i$ (30), but we retain these three elements when computing matrix-vectors products with matrix K (25). The matrices $\widetilde{H}_i$ are decomposed by the routine DGBTRF of LAPACK for banded matrices [1]. We consider the value $N = 1000$, parameters $\alpha = 1, \beta = 1$, time interval $[t_0, t_{\text{end}}] = [0, 2]$, the Radau IIA coefficients, and the GMRES linear iterative method with a restart parameter of $m = 20$. We have taken the absolute and relative error tolerances for each component equal to the same error tolerance TOL. We give some statistics obtained with the code SPARK3 on this problem in Table 2. Since this problem is linear, approximately only one Newton iteration per timestep was taken.

For the second experiment we consider a reaction-diffusion problem, the *Brusselator* system in one spatial variable (see [24]),

$$\frac{\partial u}{\partial t} = A + u^2 v - (B+1)u + \alpha \frac{\partial^2 u}{\partial x^2},$$
$$\frac{\partial v}{\partial t} = Bu - u^2 v + \alpha \frac{\partial^2 v}{\partial x^2},$$

where $x \in [0, 1]$ and $\alpha \geq 0, A$, and B are constant parameters. The boundary conditions for u and v are $u(0, t) = 1 = u(1, t), v(0, t) = 3 = v(1, t), u(x, 0) = 1 + \sin(2\pi x), v(x, 0) = 3$. We apply the method of lines by discretizing the diffusion terms using finite differences on a grid of N points $x_i = i/(N+1)$ for $i = 1, \ldots, N$,

Table 3
*Results of SPARK*3 *on the Brusselator equations* (35).

Error tolerance TOL	10^{-3}	10^{-6}	10^{-9}	10^{-12}
TOL-norm error at t_{end}	0.37	0.53	0.21	0.08
CPU-time [s]	1.02	2.06	6.13	28.71
number of steps	21	43	187	1021
number of rejected steps	4	4	1	0
number of function evaluations	195	369	1128	6144
number of inexact simplified Newton iterations	65	123	376	2048
number of J evaluations	13	35	76	33
number of P decompositions	21	43	85	61
number of P solves	127	244	751	4088
number of linear iterations	62	121	375	2040
number of matrix-vector products	62	121	375	2040

Table 4
*Results of SPARK*3 *on the Brusselator equations* (35) *with the exact simplified Newton method.*

Error tolerance TOL	10^{-3}	10^{-6}	10^{-9}	10^{-12}
TOL-norm error at t_{end}	0.67	0.58	0.21	0.08
number of steps	19	47	188	1020
number of rejected steps	5	6	2	0
number of function evaluations	183	387	1128	6111
number of simplified Newton iterations	59	129	376	2037

$\Delta x = 1/(N+1)$. We consider the value $N = 500$ and parameters $A = 1, B = 3, \alpha = 0.02$. We obtain a large system of $2N = 1000$ differential equations

$$\text{(35a)} \qquad \frac{du_i}{dt} = 1 + u_i^2 v_i - 4u_i + \frac{0.02}{(\Delta x)^2}\left(u_{i-1} - 2u_i + u_{i+1}\right),$$

$$\text{(35b)} \qquad \frac{\partial v_i}{\partial t} = 3u_i - u_i^2 v_i + \frac{0.02}{(\Delta x)^2}\left(v_{i-1} - 2v_i + v_{i+1}\right),$$

where $u_0(t) = 1 = u_{N+1}(t)$, $v_0(t) = 3 = v_{N+1}(t)$, $u_i(0) = 1 + \sin(2\pi x_i)$, and $v_i(0) = 3$ for $i = 1, \ldots, N$. We consider the time interval $[t_0, t_{\text{end}}] = [0, 10]$, the Radau IIA coefficients, and only one Richardson iteration per simplified Newton iteration. The eigenvalues of the Jacobian matrix J have a wide spectrum. The largest negative eigenvalue of J is close to -20000, so the system is really stiff. By ordering the variables as $y = (u_1, v_1, u_2, v_2, u_3, v_3, \ldots)$, the matrices $I - \gamma h J$ have a bandwidth of 2. They are decomposed using the routine `DGBTRF` of LAPACK for banded matrices [1]. We have taken the absolute and relative error tolerances for each component equal to a certain error tolerance TOL. We give some statistics obtained with the code SPARK3 on this problem in Table 3.

In Table 4 we give some statistics for the Brusselator system using the code SPARK3 with the same parameters, except that this time the linear systems of equations of the simplified Newton method are solved up to the machine precision. We observe that for the same accuracy the number of simplified Newton iterations stays quasi-identical compared to Table 3. It is a clear numerical indication of the extreme quality of the new preconditioner.

The numerical experiments discussed in this section were made on a HP VISUALIZE workstation model C240 with a 236MHz PA-RISC 8200 processor.

10. Conclusion. We have considered the application of IRK methods to implicit systems of ODEs. The major difficulty and computational bottleneck for an efficient implementation of these numerical integration methods is to solve the resulting nonlinear systems of equations. For this purpose we have suggested the use of inexact simplified Newton methods, more precisely, of simplified Newton-iterative methods. Linear systems of the simplified Newton method are solved approximately with a preconditioned linear iterative method, such as preconditioned versions of Richardson or GMRES iterations. The preconditioner considered here is an approximate inverse of the block-LU decomposition of the simplified Jacobian after W-transformation of the RK coefficients. This technique has been implemented in the new code SPARK3 and has been shown to be effective on two problems with diffusion.

Acknowledgments. The code SPARK3 was initially developed during a visit to Professors Ernst Hairer and Gerhard Wanner, both from the Department of Mathematics of the University of Geneva, Switzerland. They are acknowledged for their support. Further development of SPARK3 was made possible at the Institute for Mathematics and Its Applications of the University of Minnesota in Minneapolis thanks to a Vacheron and Constantin research prize from the University of Geneva and to the support of professor Bernardo Cockburn from the Department of Mathematics of the University of Minnesota. The author would also like to thank Professor Rebecca Matthews for proofreading the English and Dacian Daescu for finding a few typos.

REFERENCES

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen, *LAPACK User's Guide*, 2nd ed., SIAM, Philadelphia, 1995.

[2] R. E. Bank and D. J. Rose, *Global approximate Newton methods*, Numer. Math., 37 (1981), pp. 279–295.

[3] P. N. Brown and A. C. Hindmarsh, *Matrix-free methods for stiff systems of ODE's*, SIAM J. Numer. Anal., 23 (1986), pp. 610–638.

[4] P. N. Brown, A. C. Hindmarsh, and L. R. Petzold, *Using Krylov methods in the solution of large-scale differential-algebraic systems*, SIAM J. Sci. Comput., 15 (1994), pp. 1467–1488.

[5] P. N. Brown and Y. Saad, *Hybrid Krylov methods for nonlinear systems of equations*, SIAM J. Sci. Stat. Comput., 11 (1990), pp. 450–481.

[6] S. L. Campbell, I. C. F. Ipsen, C. T. Kelley, and C. D. Meyer, *GMRES and the minimal polynomial*, BIT, 36 (1996), pp. 32–43.

[7] R. P. K. Chan, *On symmetric Runge-Kutta methods of high order*, Computing, 45 (1990), pp. 301–309.

[8] T. F. Chan and K. R. Jackson, *The use of iterative linear-equation solvers in codes for large systems of stiff IVPs for ODEs*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 378–417.

[9] G. J. Cooper and J. C. Butcher, *An iteration scheme for implicit Runge-Kutta methods*, IMA J. Numer. Anal., 3 (1983), pp. 127–140.

[10] G. J. Cooper and R. Vignesvaran, *A scheme for the implementation of implicit Runge-Kutta methods*, Computing, 45 (1990), pp. 321–332.

[11] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.

[12] S. C. Eisenstat and H. F. Walker, *Globally convergent inexact Newton method*, SIAM J. Optim., 4 (1994), pp. 393–422.

[13] S. C. Eisenstat and H. F. Walker, *Choosing the forcing terms in an inexact Newton method*, SIAM J. Sci. Comput., 17 (1996), pp. 16–32.

[14] V. Frayssé, L. Giraud, and S. Gratton, *A Set of GMRES Routines for Real and Complex Arithmetic*, Tech. Rep. PA/97/49, CERFACS, Toulouse, France, 1997.

[15] C. W. Gear and Y. Saad, *Iterative solution of linear equations in ODE codes*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 583–601.

[16] A. George, *On block elimination for sparse linear systems*, SIAM J. Numer. Anal., 11 (1974), pp. 585–603.

[17] G. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, 1996.

[18] S. González-Pinto, C. González-Concepción, and J. I. Montijano, *Iterative schemes for Gauss methods*, Comput. Math. Appl., 27 (1994), pp. 67–81.

[19] S. González-Pinto, J. I. Montijano, and L. Rández, *Iterative schemes for three-stage implicit Runge-Kutta methods*, Appl. Numer. Math., 17 (1995), pp. 363–382.

[20] A. Greenbaum, *Iterative Methods for Solving Linear Systems*, Frontiers Appl. Math. 17, SIAM, Philadelphia, 1997.

[21] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations* I. *Nonstiff problems*, 2nd ed., Springer Ser. Comput. Math. 18, Springer-Verlag, Berlin, 1993.

[22] E. Hairer and G. Wanner, *Algebraically stable and implementable Runge–Kutta methods of high order*, SIAM J. Numer. Anal., 18 (1981), pp. 1098–1108.

[23] E. Hairer and G. Wanner, *Characterization of non-linearly stable implicit Runge-Kutta methods*, in Numerical Integration of Differential Equations, Lecture Notes in Math. 968, Springer, Berlin, New York, 1982, pp. 207–219.

[24] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations* II. *Stiff and Differential-Algebraic Problems*, 2nd ed., Springer Ser. Comput. Math. 14, Springer-Verlag, Berlin, 1996.

[25] W. Hoffmann and J. J. B. de Swart, *Approximating Runge-Kutta matrices by triangular matrices*, BIT, 37 (1997), pp. 346–354.

[26] L. Jay, *Symplectic partitioned Runge–Kutta methods for constrained Hamiltonian systems*, SIAM J. Numer. Anal., 33 (1996), pp. 368–387.

[27] L. O. Jay, *Structure preservation for constrained dynamics with super partitioned additive Runge–Kutta methods*, SIAM J. Sci. Comput., 20 (1998), pp. 416–446.

[28] L. O. Jay, *User's Guide to SPARK*3, Tech. Rep., Department of Math., University of Iowa, 2000, in preparation.

[29] L. O. Jay and T. Braconnier, *A parallelizable preconditioner for the iterative solution of implicit Runge-Kutta type methods*, J. Comput. Appl. Math., 111 (1999), pp. 63–76.

[30] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, Frontiers Appl. Math. 16, SIAM, Philadelphia, 1995.

[31] W. Rheinboldt, *Methods for Solving Systems of Nonlinear Equations*, 2nd ed., CBMS-NSF Regional Conf. Ser. in Appl. Math. 70, SIAM, Philadelphia, 1998.

[32] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Co., Boston, 1996.

[33] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[34] P. J. van der Houwen and J. J. B. de Swart, *Triangularly implicit iteration methods for ODE-IVP solvers*, SIAM J. Sci. Comput., 18 (1997), pp. 41–55.