ELSEVIER

Journal of Computational and Applied Mathematics 111 (1999) 63–76

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

www.elsevier.nl/locate/cam

A parallelizable preconditioner for the iterative solution of implicit Runge–Kutta-type methods

Laurent O. Jay[a,*,1], Thierry Braconnier[b]

[a] *Institute for Mathematics and its Applications, University of Minnesota, 514 Vincent Hall, 206 Church Street S.E., Minneapolis, MN 55455, USA*

[b] *Université de la Réunion, IREMIA, Département de Mathématiques et Informatique, 15 Avenue René Cassin, BP 7151, 97715 Saint-Denis Messag, Cedex 9, La Réunion, France*

Received 28 April 1998; received in revised form 8 April 1999

Abstract

The main difficulty in the implementation of most standard implicit Runge–Kutta (IRK) methods applied to (stiff) ordinary differential equations (ODEs) is to efficiently solve the nonlinear system of equations. In this article we propose the use of a preconditioner whose decomposition cost for a parallel implementation is equivalent to the cost for the implicit Euler method. The preconditioner is based on the W-transformation of the RK coefficient matrices discovered by Hairer and Wanner. For stiff ODEs the preconditioner is by construction asymptotically exact for methods with an invertible RK coefficient matrix. The methodology is particularly useful when applied to super partitioned additive Runge–Kutta (SPARK) methods. The nonlinear system can be solved by inexact simplified Newton iterations: at each simplified Newton step the linear system can be approximately solved by an iterative method applied to the preconditioned linear system. © 1999 Published by Elsevier Science B.V. All rights reserved.

Keywords: Implicit Runge–Kutta method; Inexact simplified Newton iteration; Iterative method; Parallelism; Preconditioning; Stiffness; W-transformation

1. Introduction

This article is concerned with the implementation of implicit Runge–Kutta (IRK) methods such as those based on Gauss, Radau, and Lobatto points applied to (stiff) ordinary differential equations (ODEs) [20]. The main difficulty is to efficiently solve the nonlinear system of equations. For an

* Corresponding author.

E-mail addresses: na.ljay@na-net.ornl.gov (L.O. Jay), thierry.braconnier@univ-reunion.fr (T. Braconnier)

[1] The work of this author was supported in part by a research grant of the Holderbank Foundation, Switzerland (Holderbank-Stiftung zur Forderung der wissenschaftlichen Fortbildung).

s-stage IRK method and a differential system of dimension n, the nonlinear system is of size $s \cdot n$ and it is usually solved by simplified Newton iterations. A direct decomposition of the $s \cdot n \times s \cdot n$ simplified Jacobian matrix is generally inefficient when $s \geqslant 2$. The diagonalization of the RK coefficient matrix can drastically reduce the number of operations and it also allows for parallelism. Nevertheless, the presence in general of pairs of complex eigenvalues in the RK coefficient matrix for most standard IRK methods not only impairs their parallelism, but also significantly increases the decomposition cost of the simplified Jacobian. Moreover, if several distinct IRK methods are used in a partitioned and/or additive way, this diagonalization procedure cannot be applied since the different RK matrices generally possess distinct eigenvectors. In this article we propose the use of a preconditioner requiring s independent decompositions of submatrices of dimension n, i.e., whose decomposition cost for a parallel implementation is equivalent to the cost for the implicit Euler method. The preconditioner is based on the W-transformation of the RK coefficient matrices discovered by Hairer and Wanner [20]. For stiff ODEs the preconditioner is by construction asymptotically exact for methods with an invertible RK coefficient matrix. The methodology is particularly useful when applied to super partitioned additive Runge–Kutta (SPARK) methods [22]. The nonlinear system can be solved by inexact simplified Newton iterations: at each simplified Newton step the linear system can be approximately solved by an iterative method applied to the preconditioned linear system.

In Section 2 we give the definition of IRK methods, we discuss the approximate Jacobian matrix used in the simplified Newton iterations, and we succinctly describe the W-transformation. In Section 3 we present the preconditioner used for the solution of the linear systems occurring in the simplified Newton iterations. The preconditioner is analyzed on the scalar linear test equation $y' = \lambda y$ in Section 4. In Section 5 we show how to extend the preconditioner from IRK to SPARK methods. In Section 6 we present some numerical results illustrating the behavior of the considered preconditioner using different iterative methods.

2. IRK methods, approximate Jacobian, and W-transformation

We consider the system of (stiff) ODEs

$$y' = f(t, y), \tag{1}$$

where $y = (y^1, \dots, y^n)^{\mathrm{T}} \in \mathbf{R}^n$. The definition of IRK methods is as follows.

Definition 1. One step of an s-stage *implicit Runge–Kutta* (IRK) method applied to the system (1) with initial values y_0 at t_0 and stepsize h reads

$$\begin{aligned} &Y_i - y_0 - h\sum_{j=1}^{s} a_{ij} f(t_0 + c_j h, Y_j) = 0 \quad \text{for } i = 1, \dots, s, \\ &y_1 = y_0 + h\sum_{i=1}^{s} b_i f(t_0 + c_i h, Y_i). \end{aligned} \tag{2}$$

Eqs. (2) define a nonlinear system of dimension $s \cdot n$ to be solved for the s *internal stages* Y_i. The numerical approximation at $t_0 + h$ is then given by y_1. The RK coefficients are usually expressed using a Butcher–tableau notation

$$\begin{array}{c|ccc} c_1 & a_{11} & \cdots & a_{1s} \\ \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & \cdots & a_{ss} \\ \hline & b_1 & \cdots & b_s \end{array}$$

where $b=(b_1,\ldots,b_s)^{\mathrm{T}}$ is the *weight vector*, $c=(c_1,\ldots,c_s)^{\mathrm{T}}$ is the *node vector*, and $A=(a_{ij})_{i,j=1,\ldots,s}$ is the *RK coefficient matrix*. A detailed presentation of the construction of IRK methods can be found in [20, Section IV.5]. We will assume $s\geqslant 2$ for the remainder of the article. The nonlinear system can be solved by simplified Newton iterations with approximate Jacobian matrix

$$I_s \otimes I_n - hA \otimes J \quad \text{with } J := \frac{\partial f}{\partial y}(t_0, y_0), \tag{3}$$

where $\otimes$ is the tensor product and I_m denotes the identity matrix in $\mathbf{R}^m$. Each iteration requires the solution of an $(s\cdot n)$-dimensional linear system with matrix (3) whose direct decomposition is generally inefficient for $s\geqslant 2$ as it can be drastically improved by exploiting its special structure. By diagonalizing the RK coefficient matrix A

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

the linear system can be transformed into a decoupled linear system with block-diagonal matrix

$$S^{-1}\otimes I_n(I_s\otimes I_n - hA\otimes J)S\otimes I_n = I_s\otimes I_n - h\Lambda\otimes J = \begin{pmatrix} I_n-\lambda_1 hJ & & O \\ & \ddots & \\ O & & I_n-\lambda_s hJ \end{pmatrix}. \tag{4}$$

Nevertheless, the presence in general of pairs of complex eigenvalues in the RK coefficient matrix for most standard IRK methods, not only impairs the parallelism, but also significantly increases the decomposition cost of (4) [20, Section IV.8]. This cost for such methods should be ideally equivalent to s independent decompositions of submatrices of dimension n, as for diagonally implicit Runge–Kutta (DIRK) methods [1,10,23] and multi-implicit Runge–Kutta (MIRK) methods [2,5,24] where the eigenvalues are real. Various iteration schemes have been proposed, some of them requiring the decomposition of only one submatrix of dimension n [8,9,16,17] or of s submatrices of dimension n [21,27]. These methods do not usually iterate at the linear algebra level and they can be considered as modified Newton iterations. Unfortunately, none of these methods is asymptotically exact for stiff systems, whereas the method that we present here gives by construction an asymptotically exact result in this situation.

In this article we propose a different approach aimed at reducing the amount of computations. Instead of solving exactly a linear system at each simplified Newton step, we apply an iterative method to a corresponding preconditioned linear system. The use of iterative methods for the numerical solution of stiff ODEs was already considered in [3,7,12], with an emphasis on preconditioning in [4]. *Inexact Newton* methods are generally considered to be among the most efficient ways to solve nonlinear system of equations [11,25]. We construct here a preconditioner requiring s independent

Table 1
Missing coefficients of the transformed matrix X for some IRK methods, $\sigma = (2s-1)/(s-1)$

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

decompositions of matrices of dimension n, i.e., whose decomposition cost for a parallel implementation is equivalent to the cost for the implicit Euler method. The preconditioner is based on the W-transformation of the RK coefficient matrices discovered by Hairer and Wanner [18,19]. This transformation is given by

$$X := W^{\mathrm{T}}BAW, \tag{5}$$

where $B = \operatorname{diag}(b_1, \dots, b_s)$ and the coefficients of the matrix W are given by $w_{ij} = P_{j-1}(c_i)$ with $P_k(x)$ being the kth-shifted Legendre polynomial

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

For more details about the W-transformation we refer the reader to [20, Section IV.5,6,13]. In the remainder of the article we will assume that

$$X := W^{\mathrm{T}}BAW \text{ is tridiagonal, } D := W^{\mathrm{T}}BW \text{ is diagonal and nonsingular,}$$

two conditions which are satisfied for most IRK methods of interest, such as Gauss, Radau IA & IIA, Lobatto IIIA & IIIB & IIIC & IIIC* schemes [6,13,20,22]. For these IRK methods the transformed matrix X and the matrix D read

$$X = \begin{pmatrix} 1/2 & -\zeta_1 & & & O \\ \zeta_1 & 0 & \ddots & & \\ & \ddots & \ddots & -\zeta_{s-2} & \\ & & \zeta_{s-2} & 0 & \beta_{s-1,s} \\ O & & & \beta_{s,s-1} & \beta_{ss} \end{pmatrix}, \quad D = \operatorname{diag}(1, 1, \dots, 1, d_s), \tag{6}$$

where $\zeta_k = 1/(2\sqrt{4k^2-1})$ and the missing coefficients $\beta_{s,s-1}, \beta_{s-1,s}, \beta_{ss}, d_s$ are given in Table 1. Note that the inverse of W is simply given by $W^{-1} = D^{-1}W^{\mathrm{T}}B$. We will actually assume the specific forms (6) in the remainder of the article.

3. Preconditioning the linear system

Using the W-transformation (5) in (3), at each simplified Newton step we should solve a linear system

$$Kx = b. \tag{7}$$

for a block-tridiagonal matrix

$$K = D \otimes I_n - hX \otimes J = \begin{pmatrix} E_1 & F_1 & & & O \\ G_1 & E_2 & F_2 & & \\ & \ddots & \ddots & \ddots & \\ & & G_{s-2} & E_{s-1} & F_{s-1} \\ O & & & G_{s-1} & E_s \end{pmatrix} \tag{8}$$

with $n \times n$ blocks given as follows:

$$E_1 = I_n - \tfrac{1}{2}hJ, \quad E_i = I_n \text{ for } i = 2,\dots,s-1, \quad E_s = d_s I_n - \beta_{ss} hJ, \tag{9a}$$

$$F_i = \zeta_i hJ \quad \text{for } i = 1,\dots,s-2, \quad F_{s-1} = -\beta_{s-1,s} hJ, \tag{9b}$$

$$G_i = -\zeta_i hJ \quad \text{for } i = 1,\dots,s-2, \quad G_{s-1} = -\beta_{s,s-1} hJ. \tag{9c}$$

A way to solve (7) would be to use the block-LU decomposition [14,15] of (8)

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

where the blocks H_i are recursively given by

$$H_1 = E_1, \qquad H_i = E_i - G_{i-1} H_{i-1}^{-1} F_{i-1} \quad \text{for } i = 2,\dots,s \tag{10}$$

and are assumed to be nonsingular. Subdividing the solution vector x, the right-hand side b of (7), and an intermediate vector y into s n-dimensional subvectors

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

the linear system (7) can be solved using block forward and backward substitutions

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

From (9) and (10) the blocks H_i are given in our situation by

$$H_1 = I_n - \tfrac{1}{2}hJ, \qquad H_i = I_n + \zeta_{i-1}^2 h^2 J H_{i-1}^{-1} J \quad \text{for } i = 2,\dots,s-1,$$
$$H_s = d_s I_n - \beta_{ss} hJ - \beta_{s,s-1}\beta_{s-1,s} h^2 J H_{s-1}^{-1} J.$$

Since each block H_i for $i \geqslant 2$ depends on H_{i-1}^{-1} the above recursion is not parallelizable. Moreover, we should also assume that all blocks H_i are nonsingular, a condition which can actually be violated even if $I_n - hJ$ is supposed to be invertible for all $h \geqslant 0$. In terms of computational cost at each step i for $i \geqslant 2$ we should compute $JH_{i-1}^{-1}J$. For example if the LU decomposition of H_{i-1} would be performed, this would require $7n^3/3$ operations ($n^3/3$ operations for the LU decomposition and $2n^3$ operations for the two matrix-matrix multiplications). Thus, the total block-LU decomposition of K would require $(7s-6)n^3/3$ operations. This is clearly inefficient as it would still be a factor from 3.5 to 7 more costly than if the block-diagonal-LU decomposition of (4) would be used (3.5 at best if all the eigenvalues of the RK coefficient matrix consist only of conjugate complex pairs, 7 at worst if all those eigenvalues are real).

We now present the main idea of this article. Instead of solving (7) directly, we apply an iterative method for example to the left-preconditioned linear system

$$P^{-1}Kx = P^{-1}b. \tag{11}$$

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

$$P := \begin{pmatrix} I_n & & & & O \\ G_1\tilde{H}_1^{-1} & I_n & & & \\ & \ddots & \ddots & & \\ & & G_{s-2}\tilde{H}_{s-2}^{-1} & I_n & \\ O & & & G_{s-1}\tilde{H}_{s-1}^{-1} & I_n \end{pmatrix} \begin{pmatrix} \tilde{H}_1 & F_1 & & & O \\ & \tilde{H}_2 & F_2 & & \\ & & \ddots & \ddots & \\ & & & \tilde{H}_{s-1} & F_{s-1} \\ O & & & & \tilde{H}_s \end{pmatrix} \tag{12}$$

with

$$\tilde{H}_i := I_n - \gamma_i hJ \quad \text{for} = 1,\dots,s-1, \qquad \tilde{H}_s := d_s I_n - \gamma_s hJ, \tag{13}$$

where

$$\gamma_1 = \frac{1}{2}, \quad \gamma_i = \frac{\zeta_{i-1}^2}{\gamma_{i-1}} \text{ for } i = 2,\dots,s-1, \quad \gamma_s = \beta_{ss} - \frac{\beta_{s,s-1}\beta_{s-1,s}}{\gamma_{s-1}}. \tag{14}$$

Each $\tilde{H}_i$ can be formed and decomposed independently, making these operations fully parallelizable. The coefficients γ_i have been chosen so that $\tilde{H}_i^{-1}H_i \approx I_n$ when $(I_n - hJ)^{-1}(-hJ) \approx I_n$ for all $h \geqslant h_0 > 0$. Hence, if the RK coefficient matrix is invertible, i.e., if $\gamma_s \neq 0$, the preconditioner is asymptotically exact for stiff systems. Note that for the Lobatto IIIA, IIIB, and IIIC* coefficients we have $\gamma_s = 0$. Since the Lobatto IIIC* methods behave like explicit RK methods, they should not be considered to treat stiff terms. For the Lobatto IIIA and IIIB methods the last block is $\tilde{H}_s = d_s I_n$ and need not be decomposed.

We can interpret the above preconditioner P by obtaining an explicit expression from the above formulas and we get

$$P = \begin{pmatrix} \tilde{E}_1 & F_1 & & & O \\ G_1 & \tilde{E}_2 & F_2 & & \\ & \ddots & \ddots & \ddots & \\ & & G_{s-2} & \tilde{E}_{s-1} & F_{s-1} \\ O & & & G_{s-1} & \tilde{E}_s \end{pmatrix},$$

where the blocks $\tilde{E}_i$ are recursively given by

$$\tilde{E}_1 = \tilde{H}_1 = H_1 = E_1, \qquad \tilde{E}_i = \tilde{H}_i + G_{i-1}\tilde{H}_{i-1}^{-1}F_{i-1} \quad \text{for } i = 2, \ldots, s.$$

We note that the above preconditioner is consistent in the sense that for $h=0$ we have $P=K=I_s \otimes I_n$. It is also asymptotically exact for $y' = \lambda y$ when $|h\lambda| \to \infty$ if $\gamma_s \neq 0$ (see also next section). We would like to stress the point that this preconditioner cannot be interpreted as being of the form

$$I_s \otimes I_n - h\tilde{A} \otimes J$$

for a modified coefficient matrix $\tilde{A}$ as considered in [21,27]. Note that the Jacobian matrix J in (13) can be replaced by an approximation $\hat{J}$ provided $I_n - h\bar{J}$ is a good preconditioner to $I_n - hJ$. Actually the Jacobian matrix J can also be directly replaced by an approximation $\hat{J}$ in the original system of linear equations (3), hence also in (8), (9), etc.

4. Linear analysis of the preconditioner

We consider the scalar linear test equation $y' = \lambda y$ and we denote $z := h\lambda$. The preconditioner presented in the previous section is by construction exact for $z = 0$ and asymptotically exact for large $|z|$ if $\gamma_s \neq 0$, i.e., $P^{-1}(z)K(z) \to I_s$ when $|z| \to \infty$, with $P(z)$ and $K(z)$ given below. Here, we consider intermediate values $\mathrm{Re}(z) \leqslant 0$. The matrix $K(z)$ (8) is given by

$$K(z) = \begin{pmatrix} 1 - z/2 & \zeta_1 z & & & O \\ -\zeta_1 z & 1 & \ddots & & \\ & \ddots & \ddots & \zeta_{s-2} z & \\ & & -\zeta_{s-2} z & 1 & -\beta_{s-1,s} z \\ O & & & -\beta_{s,s-1} z & d_s - \beta_{ss} z \end{pmatrix}.$$

The left-preconditioned linear system (11) reads

$$P^{-1}(z)K(z)x = P^{-1}(z)b,$$

where

$$P(z) = \begin{pmatrix} 1 & & & & O \\ g_1(z)\tilde{h}_1^{-1}(z) & 1 & & & \\ & \ddots & \ddots & & \\ & & g_{s-2}(z)\tilde{h}_{s-2}^{-1}(z) & 1 & \\ O & & & g_{s-1}(z)\tilde{h}_{s-1}^{-1}(z) & 1 \end{pmatrix} \times$$

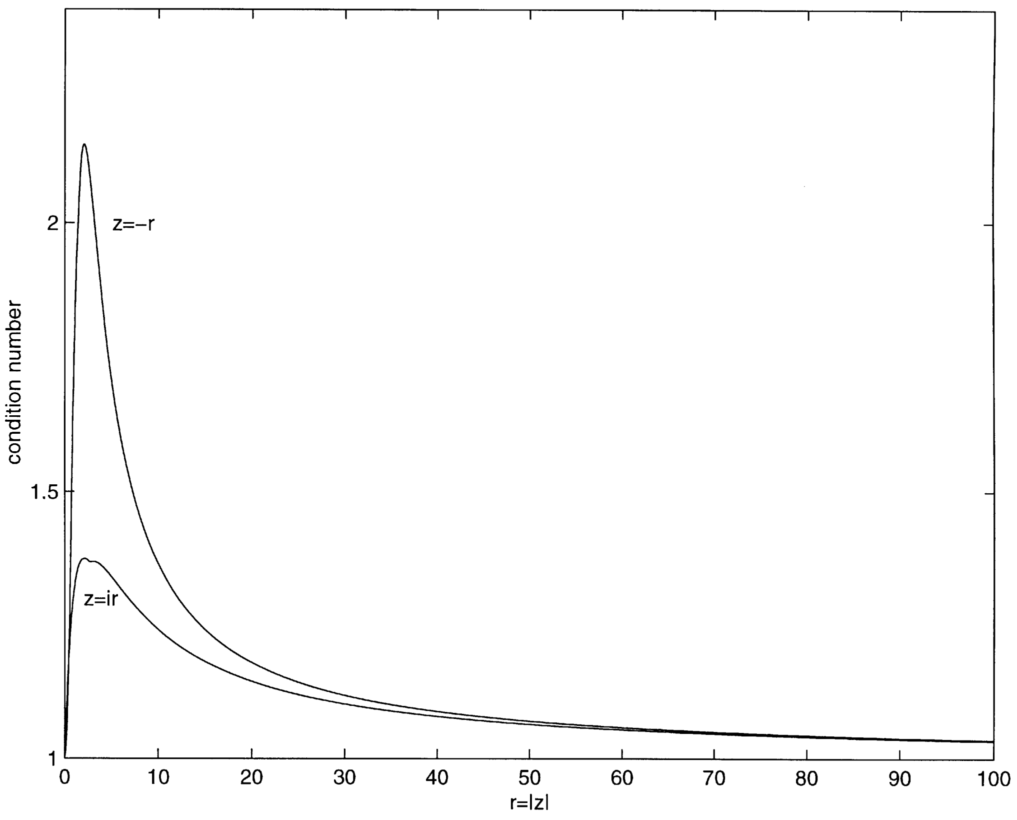

Fig. 1. Condition number $\kappa_\infty(P^{-1}(z)K(z))$ for the 2-stage Lobatto IIIC method.

$$\begin{pmatrix} \tilde{h}_1(z) & f_1(z) & & & O \\ & \tilde{h}_2(z) & f_2(z) & & \\ & & \ddots & \ddots & \\ & & & \tilde{h}_{s-1}(z) & f_{s-1}(z) \\ O & & & & \tilde{h}_s(z) \end{pmatrix}$$

$$= \begin{pmatrix} \tilde{e}_1(z) & f_1(z) & & & O \\ g_1(z) & \tilde{e}_2(z) & \ddots & & \\ & \ddots & \ddots & f_{s-2}(z) & \\ & & g_{s-2}(z) & \tilde{e}_{s-1}(z) & f_{s-1}(z) \\ O & & & g_{s-1}(z) & \tilde{e}_s(z) \end{pmatrix}$$

with function coefficients given by

$$f_i(z) = \zeta_i z \quad \text{for } i = 1, \ldots, s-2, \quad f_{s-1}(z) = -\beta_{s-1,s} z,$$
$$g_i(z) = -\zeta_i z \quad \text{for } i = 1, \ldots, s-2, \quad g_{s-1}(z) = -\beta_{s,s-1} z,$$

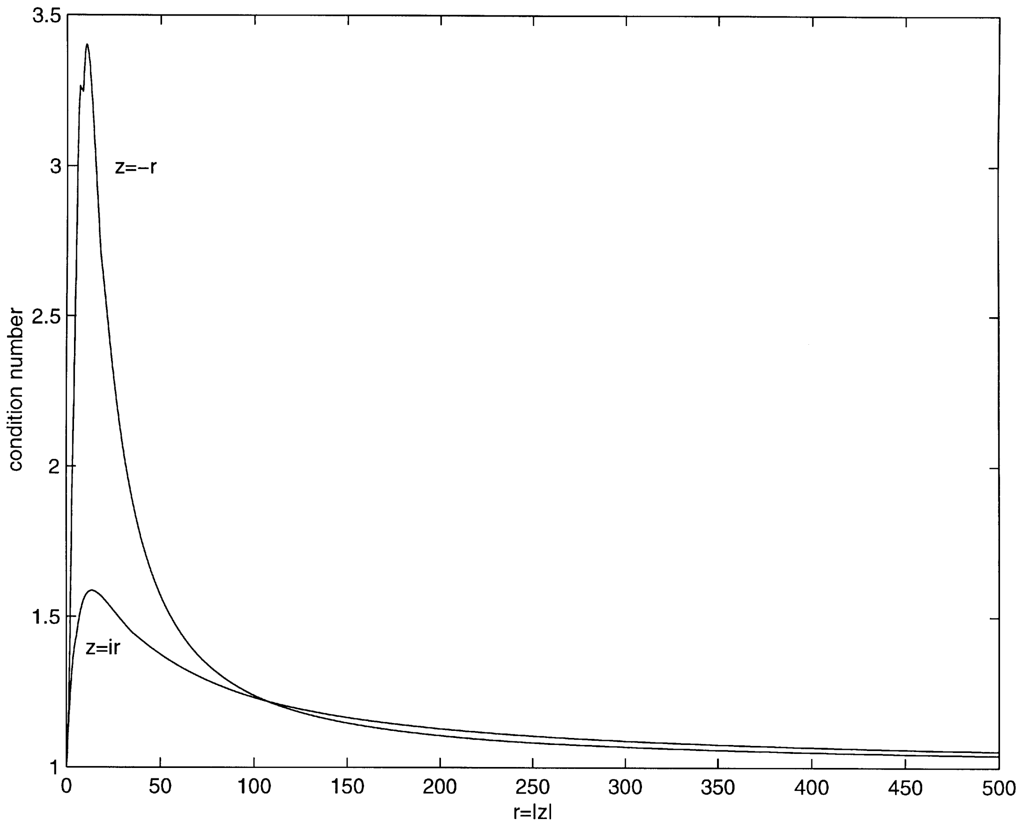

Fig. 2. Condition number $\kappa_\infty(P^{-1}(z)K(z))$ for the 5-stage Lobatto IIIB method.

$$\tilde{h}_i(z) = 1 - \gamma_i z \quad \text{for } i = 1,\dots,s-1, \quad \tilde{h}_s(z) = d_s - \gamma_s z,$$

$$\tilde{e}_1 = \tilde{h}_1(z), \quad \tilde{e}_i(z) = \tilde{h}_i(z) + g_{i-1}(z)\tilde{h}_{i-1}^{-1}(z) f_{i-1}(z) \quad \text{for } i = 2,\dots,s.$$

The quality of the preconditioner can be measured by $\kappa(P^{-1}(z)K(z))$ where κ denotes the condition number of a matrix. The closer to one this quantity is, the better the preconditioner is. For the 2-stage Lobatto IIIA and IIIB methods we trivially have $\kappa(P^{-1}(z)K(z)) = 1$ since the preconditioner is exact for those two methods, i.e., $P = K$ for any J. For the 2-stage Lobatto IIIC method we have

$$P^{-1}(z)K(z) = \begin{pmatrix} 1 & \frac{\sqrt{3}z^2}{(z-1)(z-2)^2} \\ 0 & 1 + \frac{z}{(z-1)(z-2)} \end{pmatrix}.$$

Hence, we get

$$\kappa_\infty(P^{-1}(z)K(z)) = \max\left(1 + \frac{\sqrt{3}|z|^2}{|z-1|\cdot|z-2|^2}, \left|1 + \frac{z}{(z-1)(z-2)}\right|\right)$$
$$\times \max\left(1 + \frac{\sqrt{3}|z|^2}{|z-2|\cdot|z^2-2z+2|}, \frac{|z-1|\cdot|z-2|}{|z^2-2z+2|}\right).$$

For $|z| \to \infty$ we thus have $\kappa_\infty(P^{-1}(z)K(z)) = 1 + 2\sqrt{3}/|z| + O(1/|z|^2)$. In Fig. 1 we have plotted this condition number for purely negative values $z = -r$ and purely imaginary values $z = \mathrm{i}r$ with $r = |z|$. In Fig. 2 we give a similar plot for the 5-stage Lobatto IIIB method.

5. A preconditioner for SPARK methods

The methodology described in this article to solve the nonlinear system of equations for IRK methods is particularly useful when considering SPARK methods such as the Lobatto IIIA-B-C-C* methods [22]. In this section we consider the following system of (stiff) ODEs:

$$y' = f(t, y) = \sum_{m=1}^{M} f^m(t, y), \tag{15}$$

where $y = (y^1, \ldots, y^n)^T \in \mathbf{R}^n$. Such a decomposition $\sum_{m=1}^{M} f^m(t, y)$ may come from a splitting and/or a partitioning of $f(t, y)$ into different terms. The functions $f^m(t, y)$ are supposed to have distinct properties and may therefore be numerically treated differently. Several motivations were given in [22] to introduce a more general class of methods than IRK methods. The definition of SPARK methods applied to (15) is as follows.

Definition 2. One step of an *s-stage super partitioned and additive Runge–Kutta (SPARK) method*, based on the same underlying quadrature formula $(b_i, c_i)_{i=1}^s$, applied to system (15) with initial value y_0 at t_0 and stepsize h reads

$$\begin{aligned} &Y_i - y_0 - h \sum_{m=1}^{M} \sum_{j=1}^{s} a_{ij}^{(m)} f^m(t_0 + c_j h, Y_j) = 0 \quad \text{for } i = 1, \ldots, s, \\ &y_1 = y_0 + h \sum_{i=1}^{s} b_i f(t_0 + c_i h, Y_i). \end{aligned} \tag{16}$$

Eqs. (16) define a nonlinear system of dimension $s \cdot n$ to be solved for the s *internal stages* Y_i. The numerical approximation at $t_0 + h$ is then given by y_1. This system can be solved by simplified Newton iterations with approximate Jacobian matrix

$$I_s \otimes I_n - h \sum_{m=1}^{L} A^{(m)} \otimes J^m \quad \text{with } J^m := \frac{\partial f^m}{\partial y}(t_0, y_0) \quad \text{for } m = 1, \ldots, L, \tag{17}$$

where $L \leqslant M$ and the stiff terms are treated with the L first RK methods. Since in general the RK coefficient matrices $A^{(m)}$ possess distinct eigenvectors the diagonalization procedure can not be applied. However, using the same W-transformation for all RK coefficient matrices we can assume that

$$X^{(m)} := W^{\mathrm{T}}BA^{(m)}W = \begin{pmatrix} 1/2 & -\zeta_1 & & & O \\ \zeta_1 & 0 & \ddots & & \\ & \ddots & \ddots & -\zeta_{s-2} & \\ & & \zeta_{s-2} & 0 & \beta_{s-1,s}^{(m)} \\ O & & & \beta_{s,s-1}^{(m)} & \beta_{ss}^{(m)} \end{pmatrix}$$

for possibly distinct $\beta_{s,s-1}^{(m)}, \beta_{s-1,s}^{(m)}, \beta_{ss}^{(m)}$, an assumption which is satisfied for the Lobatto IIIA, IIIB, and IIIC coefficients ($L = 3$) of the Lobatto IIIA-B-C-C* methods ($M = 4$). We thus obtain a block-tridiagonal matrix (8) with $n \times n$ blocks given by (9) for $J := \sum_{m=1}^{L} J^m$ except for the following blocks:

$$E_s = d_s I_n - \sum_{m=1}^{L} \beta_{ss}^{(m)} hJ^m, \quad F_{s-1} = -\sum_{m=1}^{L} \beta_{s-1,s}^{(m)} hJ^m, \quad G_{s-1} = -\sum_{m=1}^{L} \beta_{s,s-1}^{(m)} hJ^m.$$

As described in Sections 3 for IRK methods we can solve (7) using an iterative method on the left-preconditioned linear system (11) with the preconditioner P given by (12). The blocks $\widetilde{H}_i$ can be chosen as in (13) except for the last block

$$\tilde{H}_s := d_s I_n - \sum_{m=1}^{L} \gamma_s^{(m)} hJ^m \quad \text{with } \gamma_s^{(m)} = \beta_{ss}^{(m)} - \frac{\beta_{s,s-1}^{(m)} \beta_{s-1,s}^{(m)}}{\gamma_{s-1}}.$$

As mentioned before, note that we have $\gamma_s^{(m)} = 0$ for the Lobatto IIIA and IIIB coefficients.

6. Numerical results

The linear system (7) can be solved by an iterative method applied to the left-preconditioned system (11) using the preconditioner described in Section 3. Starting from $x_0 := 0$, the simplest iterative method is given by the *preconditioned Richardson iterations* (PRI)

$$x_{k+1} := (I - P^{-1}K)x_k + P^{-1}b \quad \text{for } k = 0, 1, 2, \ldots \tag{18}$$

If $\rho(I - P^{-1}K) < 1$ where ρ denotes the spectral radius of a matrix then the method converges linearly, otherwise the method diverges [15]. Another possibility is to use iterative Krylov-type methods such as the GMRES method [26]. Note that for such methods, convergence is ensured but the convergence behavior greatly depends on the spectral distribution of the matrix K or $P^{-1}K$ depending on whether the preconditioner is applied or not. In this section we illustrate the good quality of the preconditioner. All experiments in this section are done for Lobatto IIIC methods. The matrix J is set as follows

$$J := \begin{pmatrix} -\alpha & 1 & \dots & 1 \\ & -2\alpha & \ddots & \vdots \\ & & \ddots & 1 \\ O & & & -n\alpha \end{pmatrix},$$

Table 2
Some measures of the quality of the preconditioner

α	$\kappa_2(K)$	$\kappa_2(P^{-1}K)$	$\Vert K^{-1} - P^{-1}\Vert_2$	k (PRI)
10^1	5.40	3.40	9.5×10^{-1}	83
10^2	28.90	10.62	9.8×10^{-1}	77
10^3	156.80	9.53	6.1×10^{-1}	44
10^4	191.35	2.49	3.0×10^{-2}	13
10^6	194.38	1.59	1.6×10^{-4}	5
10^8	194.41	1.58	1.6×10^{-6}	3

Table 3
GMRES versus PGMRES

	T_{dec}	T_{sol}	T_{tot}	k	$\Vert\tilde{x} - x\Vert_\infty$
GMRES(5)	–	147.77	147.77	325	2.1×10^{-13}
PGMRES(5)	15.13	37.01	52.14	30	3.9×10^{-14}

where the parameter α allows to tune the size of the eigenvalues of J, hence to increase stiffness. Note that the eigenvalues of the matrix $I_n - hJ$ are all comprised in the interval $[\lambda_{\min}, \lambda_{\max}] = [1+h\alpha, 1+nh\alpha]$. All computations have been done on an SGI IRIX 5.3 workstation.

For a block size $n = 25$ and $s = 4$ blocks, Table 2 shows some results for $h = 10^{-2}$ and increasing values of α. When α increases, the eigenvalues of $I_n - hJ$ become larger and far apart from 1. Hence, the approximate submatrices $\tilde{H}_i$ become closer to the exact submatrices H_i, and therefore P^{-1} becomes a better approximation to K^{-1}. This is first illustrated in the columns labelled $\kappa_2(K)$ and $\kappa_2(P^{-1}K)$ showing that the condition number of the preconditioned matrix $P^{-1}K$ becomes closer to 1 as the parameter α increases, whereas the condition number of the original matrix K remains large. In the column labelled $\Vert K^{-1} - P^{-1}\Vert_2$ we see that P^{-1} tends to K^{-1} as α increases. In the last column labelled k (PRI) we give the number k of PRI iterations (18) to solve the system (7) for $x = (1,\dots,1)^{\mathrm{T}}$, the right-hand side being given by $b = Kx$. The error tolerance is set to $100 \cdot \varepsilon \cdot \Vert b\Vert$ where ε is the machine precision. The error is measured by $\Vert\tilde{x} - x\Vert_\infty$ where $\tilde{x}$ is the computed solution. We observe that the number k of PRI iterations decreases as the stiffness parameter α increases.

To illustrate the improvement that the use of the preconditioner can provide, we have run the non-preconditioned GMRES(m) method and the preconditioned PGMRES(m) method where m is the size of the Krylov subspace when the method is restarted. As before the exact solution is chosen to be $x = (1,\dots,1)^{\mathrm{T}}$. We have used the same type of matrix J of size $n = 200$, with $s = 10$ and parameters $h = 10^{-4}$ and $\alpha = 10^3$. As shown in Table 3 for this example, using the GMRES(5) method with the preconditioner P^{-1} roughly divides the total running time T_{tot} by a third and takes about 10 times less iterations (see column labelled k) than the unpreconditioned method. In Table 3 T_{dec} corresponds to the time in seconds to compute the decomposition of the preconditioner P, T_{sol} corresponds to the time in seconds for the resolution by the (P)GMRES(5) methods, and T_{tot} is the total computational time.

Table 4
Comparison between block-LU, PRI, and PGMRES

	T_{dec}	T_{sol}	T_{tot}	k	$\|\tilde{x}-x\|_\infty$
block-LU	1391.64	3.92	1395.56	–	2.9×10^{-14}
PRI	156.00	11.96	167.96	3	2.3×10^{-13}
PGMRES(2)	156.00	17.32	173.32	4	2.9×10^{-13}

Finally, we have applied the direct block-LU method (see (10)), PRI, and PGMRES(2) with the matrix J of size $n=500$, with $s=4$ and parameters $h=10^{-4}$ and $\alpha=10^{9}$, in order to compare the efficiency of the preconditioned iterative methods toward the direct block-LU method. Some results are shown in Table 4. T_{dec} corresponds to the time in seconds of the factorization for the direct block-LU method whereas for PRI and PGMRES(2) this corresponds to the time to decompose the preconditioner P. T_{sol} corresponds to the time in seconds for the resolution by the three different methods. The preconditioner is good since the eigenvalues of $I_n - hJ$ are large. Thus only a few iterations (see column labelled k) are needed by the two preconditioned iterative methods to solve the linear system. We can see that for the same level of accuracy, the preconditioned iterative methods take much less time than the direct method (see column labelled T_{tot}), this is simply because the computational effort needed to decompose P is much smaller than for the block-LU factorization of K. Obviously, the block-LU decomposition of the block-tridiagonal matrix K is not the optimal way to solve the system, since by diagonalizing the RK coefficient matrix we could improve the cost of the decomposition by a factor close to 4. Nevertheless, this is an interesting measure in our context since for the implementation of SPARK methods this diagonalization procedure cannot be applied. It is interesting to note that the simple PRI method is as efficient as the PGMRES(2) method. Since the direct block-LU method provides a residual $\|K\tilde{x}-b\|$ close to $C\cdot\varepsilon\cdot\|b\|$ for a constant C, for comparison reasons we have set the stopping criterion for PRI and PGMRES(2) to a similar level. For the numerical solution of (stiff) ODEs such an accuracy is not needed, because the stopping criterion within the simplified Newton iterations can be relaxed and be based on the preconditioned residual error $\|P^{-1}(K\tilde{x}-b)\|$ [4]. Moreover, nonstiff components need not be solved very accurately since the Newton iterations on top of the iterative linear solver make these components to converge sufficiently rapidly.

To conclude shortly, the new preconditioning technique proposed in this paper can save, in term of matrix decompositions and for most standard IRK methods, a factor two or more of operations over the classical approach of diagonalizing the RK coefficient matrix. Moreover, this new approach can also be easily extended to deal with SPARK methods.

References

[1] R. Alexander, Diagonally implicit Runge–Kutta methods for stiff ODEs, SIAM J. Numer. Anal. 14 (1977) 1006–1021.

[2] C. Bendtsen, Highly stable parallel Runge–Kutta methods, Appl. Numer. Math. 21 (1996) 1–8.

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

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

[5] K. Burrage, A special family of Runge–Kutta methods for solving stiff differential equations, BIT 18 (1978) 22–41.

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

[7] T.F. Chan, 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) 378–417.

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

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

[10] M. Crouzeix, Sur l'approximation des équations différentielles opérationnelles linéaires par des méthodes de Runge–Kutta, Ph.D. Thesis, Univ. Paris VI, France, 1975.

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

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

[13] S. Geng, Construction of high order symplectic Runge–Kutta methods, J. Comput. Math. 11 (1993) 250–260.

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

[15] G. Golub, C.F. van Loan, Matrix computations, Johns Hopkins Studies in the Math. Sci., 3rd Edition, The Johns Hopkins Univ. Press, Baltimore, 1996.

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

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

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

[19] E. Hairer, G. Wanner, Characterization of non-linearly stable implicit Runge–Kutta methods, in Numerical Integration of Differential Equations, Lecture Notes in Mathematics, Vol. 968, 1982, pp. 207–219.

[20] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems, 2nd Revised Edition, Comput. Math., Vol. 14, Springer, Berlin, 1996.

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

[22] L.O. Jay, Structure-preservation for constrained dynamics with super partitioned additive Runge–Kutta methods, SIAM J. Sci. Comput. 20 (1999) 416–446.

[23] S.P. Nørsett, Semi-explicit Runge–Kutta methods, Technical Report 6/74 Dept. of Math., Univ. of Trondheim, Norway, 1974.

[24] B. Orel, Parallel Runge–Kutta methods with real eigenvalues, APNUM 11 (1993) 241–250.

[25] W. Rheinboldt, Methods for Solving Systems of Nonlinear Equations, SIAM Classics in Appl. Math. SIAM, Philadelphia, Second Revised and Expanded Edition, 1999.

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

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