Proceedings of ALGORITMY 2020
pp. 11–20

NUMERICAL SOLUTION METHODS FOR IMPLICIT RUNGE-KUTTA METHODS OF ARBITRARILY HIGH ORDER

OWE AXELSSON* AND MAYA NEYTCHEVA†

Abstract. In this study we consider an efficient implementation of Implicit Runge-Kutta methods for solving large systems of ordinary differential equations that originate from finite element discretization of the heat and similar equations, to be solved on large time intervals. The main contribution of this work is to show how to implement a fully stage-parallel version of the method, utilizing the dominance of the block lower triangular part of the quadrature matrix, and to illustrate it numerically. Its usage for the solution of algebraic-differential equations is also touched.

Key words. Implicit Runge-Kutta methods, Radau methods, iterative methods, preconditioning, fully stage-parallel

AMS subject classifications. 65F10, 65D30

1. Introduction.

Evolution equations are sequential by nature. Their numerical simulation can be very time-consuming unless one provides some form of a parallelizable solution method. This is particularly important because on future computers the clock-cycle can hardly increase anymore. The computational efficiency must be coupled with the use of stable time-integration methods. Utilization of standard time-stepping methods when solving evolution equations of type (3.1) can be very costly and time-consuming and, for strongly ill-conditioned problems, such methods may not even converge. If we use an explicit time-stepping method the time-step must be chosen sufficiently small to guarantee a numerically stable solution. For ill-conditioned problems, it must often be chosen unfeasibly small.

When we use a stable implicit method such as the backward Euler, the trapezoidal method or the Crank-Nicholson method (cf. [1]), the time-step must still be small to get a sufficiently small time-integration error. In addition, for the explicit methods, one must solve systems with some mass matrix M at each time-step or for the backward Euler and trapezoidal implicit methods one must solve systems with $M + \tau K$, K being a stiffness matrix, or similar, where τ is the time-step. Therefore, there are strong reasons to use higher order time-integration methods, which can enable the usage of much fewer time-steps. As is well known, see e.g [2], classical multistep methods can not have an order of approximation higher than two, otherwise they are not stable for all eigenvalues of the evolution operator $M^{-1}K$ with eigenvalues in the whole right half of the complex plane, that is, they are not A-stable. This can be a severe limitation because in many problems, there can appear rapidly changing oscillations, leading to the appearance of such widespread eigenvalues. On the other hand, in [3], see also [4], it was early proven that there exist implicit Runge-Kutta methods of an arbitrarily high order that are A-stable.

Implicit Runge-Kutta methods were first presented in [5], giving the methods the name 'IRK'. Independently, in [6], such methods were presented and it was shown that due to their high order of approximation and stability properties, they could be

*The Czech Academy of Sciences, Institute of Geonics, Ostrava, Czech Republic and Department of Information Technology, Uppsala University, Uppsala, Sweden (owe.axelsson@it.uu.se)

†Department of Information Technology, Uppsala University, Uppsala, Sweden (maya.neytcheva@it.uu.se)

considered as global integration methods, that is, it could suffice to use just one or very few time-steps, i.e. very large time-step intervals. In these original papers the A-stability property of the methods is not shown, but it is shown later in [3, 7].

As is described in Section 2, there are several versions of IRK methods. All these methods are A-stable, but only the Radau method is strongly A-stable (also called strongly L-stable [8]), which means that the corresponding recursion stability factor converges to zero when the absolute values of the eigenvalues converge to infinity. The two other major methods in use, Gauss and Lobatto integration, are not strongly A-stable because for them the absolute value of the stability function converges to unity. In the presence of large eigenvalues of K or the Jacobian matrix, this implies at least a linear growth of rounding errors with increasing number of repeated time steps. Furthermore, the Radau method does not suffer from order reduction, which can occur for instance in the solution of systems of differential-algebraic equations, see e.g., [9, 10]. For these reasons, in this paper we consider mainly the Radau methods.

There is a practical problem with applying IRK methods, namely, that a large scale block matrix system of order $qn \times qn$ arises, where q is the stage order of the method, that is, equals to the degree and number of zeros of the quadrature polynomials.

The solution of this system can be costly and somewhat involved, which has been the major reason why IRK methods are less often used. In practice, the system must be solved by some preconditioned iterative method. Preferably, it should be possible to implement the methods efficiently in a parallel computer environment. Construction of such preconditioning methods is the major topic of this paper.

In [6], it is shown that the algebraic system, resulting from the IRK method, has a dominating lower block-diagonal (including the diagonal) part, that is, the upper off-diagonal entries are relatively small. This implies that an efficient preconditioner can be built using the block lower-triangular part of the system matrix. However, even though the latter permits some parallel computations, the solution of such block matrix system must take place sequentially, block-row by block-row, see Section 3 for further details. In this paper we propose an alternative method which is fully stage-parallel, that is, one can solve the q arising systems independently utilizing q parallel processes or groups of processes.

For earlier discussions of solution methods for the IRK methods, see [11, 12, 13] and also [14, 15, 16], where diagonally implicit Runge-Kutta (DIRK) methods are presented. DIRK methods allow parallel implementation but have a much lower order of approximation, compared with the full IRK methods and therefore still force the use of smaller time-steps. Since some time an alternative approach to solve time-dependent partial differential equations has been used (see, e.g., [17, 18, 19]), based on combined time-space finite elements (FE). This means that, e.g., a 3D space partial differential operator is solved on a 4D space-time FE mesh. Clearly, the implementation of the method is more involved but such methods enable the use of adaptive mesh resolution methods in both time and space. In this paper we apply FE only to the space domain.

We begin, in Section 2, by introducing the polynomials used for defining the numerical integration points. Section 3 presents the IRK methods of Radau type, Section 4 deals with a discussion on the use of IRK methods for differential algebraic systems and Section 5 contains numerical illustrations.

2. Quadrature polynomials. Let $Q_q(x) = \frac{1}{2^q q!}(x^2-1)^q$ and $D^q = \frac{d}{dx^q}$, $q = 1, 2, \cdots$. Note that $DQ_q(x) = xQ_{q-1}(x)$. Then the Gauss integration polynomial equals $\mathcal{P}_q(x) = D^q Q_q(x)$ and the Gauss-Radau polynomial equals

$$\mathcal{P}_q(x) - \mathcal{P}_{q-1}(x) = D^{q-1}(DQ_q(x) - Q_{q-1}(x)) = D^{q-1}((x-1)Q_{q-1}(x)).$$

It has been shown in [20] that the zeros $\tilde{c}_i$, $i = 1, \cdots, q$ of $\mathcal{P}_q(x) + a\mathcal{P}_{q-1}(x) + b\mathcal{P}_{q-2}(x)$, where $b \leq 0$, are real, distinct and located in the interval $[-1, 1]$. Further, the quadrature coefficients $\tilde{a}_{qk} = \int_{-1}^{1} \ell_k(z)\, dz$ are positive, where

$$\ell_k(z) = \prod_{i=1, i\neq k}^{q} (z - \tilde{c}_i) / \prod_{i=1, i\neq k}^{q} (\tilde{c}_k - \tilde{c}_i)$$

are the Lagrange interpolation polynomials. It follows that

$$\begin{aligned}
\mathcal{P}_q(x) - \mathcal{P}_{q-1}(x) &= D^{q-2}(Q_{q-1}(x) + x(x-1)Q_{q-2}(x)) \\
&= D^{q-3}((3x-1)Q_{q-2}(x) + x^2(x-1)Q_{q-3}(x)) \\
&= D^{q-4}(3Q_{q-2} + (6x^2 - 3x)Q_{q-3}(x) + x^3(x-1)Q_{q-4}(x)) \\
&= D^{q-5}((15x-3)Q_{q-3}(x) + (10x^3 - 6x^2)Q_{q-4}(x) + x^4(x-1)Q_{q-5}(x))
\end{aligned}$$

etc. For $q = 2, 3$ we obtain

$$\begin{aligned}
\mathcal{P}_2(x) - \mathcal{P}_1(x) &= \frac{1}{2}(x^2 - 1) + x(x-1) = \frac{1}{2}(3x+1)(x-1), \\
\mathcal{P}_3(x) - \mathcal{P}_2(x) &= \frac{5}{2}\Big(x - \frac{1}{5} - \frac{\sqrt{6}}{5}\Big)\Big(x - \frac{1}{5} + \frac{\sqrt{6}}{5}\Big)(x-1).
\end{aligned}$$

These polynomials are given for the interval $(-1, 1]$. By replacing x by $2x - 1$, we transform them to obtain integration points in the interval $(0, 1]$. For $q = 2, 3$, the transformed polynomials take the form

$$\begin{aligned}
\widetilde{\mathcal{P}}_2(x) - \widetilde{\mathcal{P}}_1(x) &= (3(2x-1)+1)(x-1) = 2(3x-1)(x-1), \\
\widetilde{\mathcal{P}}_3(x) - \widetilde{\mathcal{P}}_2(x) &= 20\left(x - \frac{3 + \sqrt{3/2}}{5}\right)\left(x - \frac{3 - \sqrt{3/2}}{5}\right)(x-1).
\end{aligned}$$

The quadrature formula $\int_0^1 f(z)\, dz = \sum_{k=1}^{q} a_{q_k} f(c_k) + \mathcal{R}_q(f)$, where $c_k = 2\tilde{c}_k - 1$ are contained in the interval $[0, 1]$ is exact, i.e. $\mathcal{R}_q(f) = 0$ for $f \in \Pi_{2q-1}$ if $b = 0$, i.e. for the Radau integration method.

3. Implicit Runge-Kutta methods of Radau type. The general form of a q-stage IRK method to solve the equation

$$y'(t) = f(t, y(t)), \quad y(0) = y_0, \quad t \in [0, T]$$

has the form

$$\begin{aligned}
y_{n+1} &= y_n + \tau \sum_{k=1}^{q} b_k Y_k, \\
Y_i &= y_n + \tau \sum_{k=1}^{q} a_{ik} f(t_n + c_k \tau Y_k), \quad i = 1, \cdots, q-1,
\end{aligned}$$

where τ is the time step, q is the number of stages, Y_k are referred to as the stage values. Here c_i, $0 < c_i \leq 1$ are the quadrature points, $a_{ik} = \int_0^{c_i} \ell_k(z)\, dz$ are the corresponding quadrature coefficients, where $i, k = 1, \ldots, q$, and $b_k = a_{qk}$. Since

$$\sum_{k=1}^{q} a_{ik} a_k^{\ell-1} = \int_0^{c_i} t^{\ell-1}\, dt = \frac{1}{\ell} c_i^\ell, \qquad i, l = 1, \cdots, q,$$

it follows that $A_q V = CVR$, where $A_q = [a_{ik}]_{i,k=1}^q$ is the full IRK quadrature matrix and $C = \text{diag}\{c_1, c_2, \cdots, c_q\}$, $R = \text{diag}\{1, 1/2, \cdots, 1/q\}$ and V is the Vandermonde matrix, generated by c_i, i.e., $V = \begin{bmatrix} 1 & c_1 & \cdots & c_1^{q-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & c_q & \cdots & c_q^{q-1} \end{bmatrix}$. Since the zeros $\{c_i\}$ are distinct, V is nonsingular. It follows that $A_q[c_i^\ell] = \frac{c_i}{\ell}[c_i^\ell], \quad i = 1, \cdots, q$, that is, $[c_i^\ell]$, $\ell = 1, \ldots, q$ are the eigenvectors of A_q and A_q^{-1} for eigenvalues c_i/ℓ, respectively ℓ/c_i.

The target evolutionary equation to be solved is

$$M\frac{d\boldsymbol{u}(t)}{dt} + K\boldsymbol{u}(t) = \boldsymbol{f}(t), t \in (0, T], \quad \boldsymbol{u}(0) = \boldsymbol{u}_0, \tag{3.1}$$

which arises from the heat equation, discretized in space using the FE method, thus, M is the mass matrix, K is the corresponding stiffness matrix, in general ill-conditioned. Both M and K may depend on the time variable. Similar equations arise in network problems and in time-harmonic Maxwell's equations where rapidly changing oscillations can occur and the eigenvalues of $M^{-1}K$ can be widespread in the right half complex plane, cf. [21], and the references therein.

Using tensor notations, in the linear case, the system we have to solve in (3.1) at each time step can be written in a Kronecker product form as (cf., e.g., [22], where a viscous wave equation has been considered)

$$(I_q \otimes M + \tau A_q \otimes K)\boldsymbol{v} = I_q \otimes \boldsymbol{f} - (I_q \otimes K)(\boldsymbol{e}_q \otimes \boldsymbol{u}_0), \tag{3.2}$$

where A_q is the IRK matrix, I_q is the identity matrix of order q and $\boldsymbol{e}_q$ is a vector of length q with all components 1.

Let the matrices M and K be real of size $n \times n$. Then the matrix $I_q \otimes M + \tau A_q \otimes K$ is of size $qn \times qn$ and the question how to efficiently solve systems with it is the major focus of this work. Clearly, the need to solve such large systems by iterative methods is indisputable, as well as the demand for a numerically and computationally efficient preconditioner.

There are different options to approach the problem. One way, suggested in e.g. [11] is to transform (3.2) to

$$\mathcal{A}\boldsymbol{w} \equiv (A_q^{-1} \otimes M + \tau I_q \otimes K)\boldsymbol{w} = A_q^{-1} \otimes \boldsymbol{f} - (A_q^{-1} \otimes K)(\boldsymbol{e}_q \otimes \boldsymbol{u}_0). \tag{3.3}$$

by letting $\boldsymbol{w} = A_q \otimes I_n \boldsymbol{v}$ with I_n being the identity matrix of size n, and utilizing the relation $(a \otimes b)(c \otimes d) = (ac) \otimes (bd)$. We observe that the second matrix term in $\mathcal{A}$ is block-diagonal. We also note, that the lower-triangular part of both A_q and A_q^{-1}, are dominating. This leads to suggesting to consider

$$\mathcal{P} = L_q \otimes M + \tau I_q \otimes K,$$

as a preconditioner to $\mathcal{A}$, where L_q is the lower-triangular part of A_q^{-1}, or its upper-Hessenberg part, including the lower-triangular plus the first upper diagonal, scaled so that the eigenvalues of L_q are real. Clearly L_q will have eigenvectors close to $[c_i^\ell]$, i.e. those of A_q^{-1}, so it holds approximately that

$$L_q T = T\Lambda \quad \text{for} \quad T = [c_i^1, c_i^2, \cdots, c_i^q].$$

Since the eigenvalues of A_q^{-1} and of the lower block triangular part of A_q^{-1} are real, one can expect that the eigenvalues of L_q are also real.

The next task is to transform $\mathcal{P}$ into a block-diagonal structure, that allows for good parallelization. Using the relation $(a\otimes b)(c\otimes d) = (ac)\otimes(bd)$, we can now easily transform the solution of $\mathcal{P}\boldsymbol{v} = \widetilde{\boldsymbol{f}}$, where $\widetilde{\boldsymbol{f}} = (T\otimes I_n)(T^{-1}\otimes I_n)\boldsymbol{f}$ to a solution with a block-diagonal matrix. Namely,

$$\begin{aligned}\mathcal{P} &= L_q\otimes M + \tau I_q\otimes K = T\Lambda T^{-1}\otimes M + \tau T T^{-1}\otimes K\\ &= (T\otimes I_n)(\Lambda\otimes M)(T^{-1}\otimes I_n) + \tau(T\otimes I_n)(I_q\otimes K)(T^{-1}\otimes I_n)\\ &= (T\otimes I_n)\left((\Lambda\otimes M) + \tau(I_q\otimes K)\right)(T^{-1}\otimes I_n) = (T\otimes I_n)\,\mathcal{P}_d\,(T^{-1}\otimes I_n),\end{aligned}$$

where $\mathcal{P}_d = \Lambda\otimes M + \tau(I_q\otimes K)$. Hence, $\mathcal{P}_d$ is block-diagonal. One can show that the vectors in T can be computed using a simple recursion.

Consider the case when L_q is the above-described upper Hessenberg part of A_q^{-1}. Then, $A_qL_q = I_q - A_qE$, where $E = A_q^{-1} - L_q$. We note that A_q has a dominating lower triangular part and E is upper triangular with alternating signs in the subdiagonals with small entries. Hence A_qE has also small entries. The following holds: $A_qL_q\boldsymbol{x} = \lambda\boldsymbol{x}$, thus, $(I_q - A_qE)\boldsymbol{x} = \lambda\boldsymbol{x}$, or $(1-\lambda)\boldsymbol{x}^T\boldsymbol{x} = \boldsymbol{x}^TA_qE\boldsymbol{x}$, which implies

$\mathrm{Re}((1-\lambda)\boldsymbol{x}^T\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T(A_qE + E^TA_q^T)\boldsymbol{x}$, $\mathrm{Im}((1-\lambda)\boldsymbol{x}^*\boldsymbol{x}) = \frac{i}{2}\boldsymbol{x}^*(A_qE - E^TA_q^T)\boldsymbol{x}$. If $\boldsymbol{x} = \boldsymbol{x}_1 + i\boldsymbol{x}_2$, where $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ are real, then $\mathrm{Im}(1-\lambda)\boldsymbol{x}^*\boldsymbol{x} = \frac{1}{2}\,i\,\boldsymbol{x}_1^T(A_qE - E^TA_q^T)\boldsymbol{x}_2$, so even if there is an imaginary part for some eigenvalues, it is small. This implies a faster rate of convergence of the iterative method.

4. Differential-algebraic system.

Consider first a stiff differential equation of singular perturbation type,

$$\begin{cases} u' &= f(u,v)\\ \varepsilon v' &= g(u,v)\end{cases}$$

where $\varepsilon > 0$ is a small parameter and the Jacobian matrix has eigenvalues with negative real part. As shown in [23] and [9], the order of approximation reduces to $u(t_m) - u_m = O(\tau^{q+1})$ for the Gauss and Lobatto integration methods but not for the Radau method, where $u(t_m) - u_m = O(\tau^{2q-1}) + O(\varepsilon^2\tau^q)$. When $\varepsilon \to 0$, we obtain a differential-algebraic equation.

A basic type of a differential-algebraic system is illustrated by the Darcy equation for porous media problems, see e.g., [24, 23], which in time dependent form is

$$\begin{cases} \mathcal{K}^{-1}\boldsymbol{v} + \boldsymbol{\nabla} p &= 0,\\ \boldsymbol{\nabla}\cdot\boldsymbol{v} + \xi\frac{\partial p}{\partial t} &= Q\end{cases} \tag{4.1}$$

It involves two physical fields, the Darcy velocity $\boldsymbol{v}$ and the fluid (pore) pressure p, which have to be computed in a domain Ω. The parameter matrix $\mathcal{K}$ is defined by permeabilities, scaled by dynamic viscosity, the coefficient ξ is related to the effective compressibility and Q stands for fluid source/sink terms. The time-dependent form in (4.1) corresponds to a flow retardation mechanism, involving small compressibility of the fluid and deformation of the porous matrix.

The discretization of the problem leads to a saddle point system involving a time-discretization method which must be stable. As has been shown in [9], when solving systems of differential-algebraic equations, for instance by Gauss or Gauss-Lobatto methods, which are not L-stable, an order reduction can occur. This is another reason for choosing the Radau time integration method.

As shown, e.g., in [24], the general form of a DAE system is

$$\mathcal{A}_1 \frac{\partial}{\partial t} U + \mathcal{A}_0 U = F, \tag{4.2}$$

where, in standard FEM notations, $\mathcal{A}_1 = \begin{bmatrix} 0 & 0 \\ 0 & -C \end{bmatrix}$, $\mathcal{A}_0 = \begin{bmatrix} M & B^T \\ B & 0 \end{bmatrix}$, $U = \begin{bmatrix} \boldsymbol{v} \\ p \end{bmatrix}$,
$\langle Mv, z\rangle = \int_\Omega \mathcal{K}^{-1} v_h z_h,\ \forall v_h, z_h \in V_h,\quad \langle M_p p, q\rangle = \int_\Omega p_h q_h,\ \forall p_h, q_h \in P_h,$
$\langle Bv, q\rangle = \int_\Omega \boldsymbol{\nabla} v_h q_h,\ \forall v_h \in V_h,\ q_h \in P_h$ and $C = c_\xi M_p$.

Assuming that $\mathcal{A}_1$ is constant, for each time-interval $[0, T]$, (4.2) results in

$$\int_0^T \mathcal{A}_1 \frac{\partial}{\partial t} U\, dt + \int_0^T (\mathcal{A}_0 U - F)\, dt = \mathcal{A}_1(U^1 - U^0) + \int_0^T (\mathcal{A}_0 U - F)\, dt = 0.$$

For the mentioned reasons of stability, the time integral must be performed by a suitable approximate integration scheme.
The simplest method is achieved for $q = 1$, i.e. with just one integration point, $c_1 = 1$,

$$\mathcal{A}_R^{(1)} U^1 := (\mathcal{A}_1 + \tau \mathcal{A}_0) U^1 = \mathcal{A}_1 U^0 + \tau F^1,$$

which is identical to the backward Euler method. For $q = 2$, the Radau method leads to the system

$$\mathcal{A}_R^{(2)} \begin{bmatrix} U^{(1/3)} \\ U^{(1)} \end{bmatrix} = \begin{bmatrix} \mathcal{A}_1 + \frac{5}{12}\tau \mathcal{A}_0 & -\frac{\tau}{12}\mathcal{A}_0 \\ \frac{3\tau}{4}\mathcal{A}_0 & \mathcal{A}_1 + \frac{\tau}{4}\mathcal{A}_0 \end{bmatrix} \begin{bmatrix} U^{(1/3)} \\ U^{(1)} \end{bmatrix} = \begin{bmatrix} \mathcal{A}_1 U^{(0)} + \frac{\tau}{12}(5F^{1/3} - F^1) \\ \mathcal{A}_1 U^{(0)} + \frac{\tau}{4}(3F^{1/3} + F^1) \end{bmatrix}. \tag{4.3}$$

Similarly to [24], we can eliminate $U^{1/3}$ to get a reduced system in a convenient form. However, to follow the general type of preconditioning method, used in the paper, we solve (4.3) using the preconditioner

$$\mathcal{B}^{(2)} = \begin{bmatrix} \mathcal{A}_1 + \frac{5}{12}\tau \mathcal{A}_0 & 0 \\ \frac{3\tau}{4}\mathcal{A}_0 & \mathcal{A}_1 + \frac{\tau}{4}\mathcal{A}_0 \end{bmatrix},$$

which can be shown to lead to a spectrum of ${\mathcal{B}^{(2)}}^{-1}\mathcal{A}_R^{(2)}$ contained in the interval $[1, \frac{8}{5}]$, see [10]. We note that each application of ${\mathcal{B}^{(2)}}^{-1}$ involves solving two systems of similar form as in the backward Euler method, but the order of local approximation errors is now $O(\tau^4)$ instead for $O(\tau^2)$ as for the Euler method.

Next, for $q = 3$, we illustrate the advantages using the lower-triangular plus possibly the next upper-diagonal of A_q^{-1} as a preconditioner. The system takes the form

$$\mathcal{A}_R^{(3)} \begin{bmatrix} U^{(c_1)} \\ U^{(c_2)} \\ U^1 \end{bmatrix} = \begin{bmatrix} \mathcal{A}_1 + \tau a_{11}\mathcal{A}_0 & \tau a_{12}\mathcal{A}_0 & \tau a_{13}\mathcal{A}_0 \\ \tau a_{12}\mathcal{A}_0 & \mathcal{A}_1 + \tau a_{22}\mathcal{A}_0 & \tau a_{23}\mathcal{A}_0 \\ \tau a_{31}\mathcal{A}_0 & \tau a_{32}\mathcal{A}_0 & \mathcal{A}_1 + \tau a_{33}\mathcal{A}_0 \end{bmatrix} \begin{bmatrix} U^{(c_1)} \\ U^{(c_2)} \\ U^1 \end{bmatrix},$$

where $c_1 = \frac{1}{5}(2 - \sqrt{3/2})$, $c_1 = \frac{1}{5}(2 + \sqrt{3/2})$ and the right hand side is as in (4.3).

Here the quadrature matrix $[a_{ik}]_{i,k=1}^3$ takes the form given in Section 3. Then,

$$L_3 A_3 = A_3^{-1} A_3 + \begin{bmatrix} 0 & 0 & -\frac{2}{5} + \frac{1\sqrt{6}}{15} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} A_3 =$$

$$= I + \begin{bmatrix} 0 & 0 & -\frac{2}{5} + \frac{4\sqrt{6}}{15} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x & x & x \\ x & x & x \\ \frac{8\sqrt{6}}{3} - 1 & -\frac{81\,1/2}{3} - 1 & 5 \end{bmatrix} I + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ x & x & -2 + \frac{4\sqrt{6}}{3} \end{bmatrix}$$

Here the distance from the unit eigenvalue is $-2 + \frac{4\sqrt{6}}{3} \approx 1.26$.

It can be seen that this method gives improved spectral values when q increases. Due to limited space, we do not consider higher order methods here.

5. Numerical tests. For simplicity, as a test problem, we use the heat equation in two space dimensions with a pre-manufactured solution of the form $u_{exact} = \sin(a\pi x)\sin(b\pi y)(1 + \sin(\pi t))e^{-c\,t}$ with $a = b = 2, c = 0.05$.

$$\begin{aligned} &\frac{\partial u}{\partial t} = \Delta u + f(x,y,t), \;\; (x,y) \in \Omega = [0,1]^2, \;\; t \in (0, 4\pi], \\ &u = 0 \text{ on } \partial\Omega, \;\; u(x,y,0) = \sin(a\pi x)\sin(b\pi y). \end{aligned} \tag{5.1}$$

The space discretization is done on a triangular mesh (depicted in Figure 5.1) with characteristic mesh size h, using bilinear FE basis functions. All numerical experiments are done in Matlab, version 9.6.0.1174912 (R2019a). For the experiments we use the form of the matrix $\mathcal{A} = A_q^{-1} \otimes M + \tau I_q \otimes K$ and a preconditioner $\mathcal{P} = L_q^{(2)} \otimes M + \tau I_q \otimes K$, reformulated as $\mathcal{P} = (T \otimes I_n)\,\mathcal{P}_d\,(T^{-1} \otimes I_n)$, $\mathcal{P}_d = (\Lambda \otimes M) + \tau(I_q \otimes K)$ and where L_q is the lower-triangular part of A_q^{-1}, decomposed as $L_q = T\Lambda T^{-1}$.

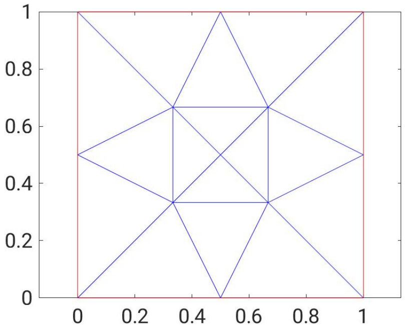

FIG. 5.1.

In order to see the behavior of the time discretization error for each chosen value of q and τ we choose h small enough, so that the discretization error in space $O(h^2)$ is less than the local discretization error in time, $O(\tau^{2q})$. The relative error, presented in Tables 5.1-5.3 is computed as $\|u_{exact} - u_{IRK}\|_2 / \|u_{exact}\|_2$.

We have performed three types of experiments.

(E1) At each time step, solve systems with the matrix $\mathcal{A}$ via a direct method.

$q = 2$		
$\tau = 2^{-2},\ h = 2^{-5},\ h^2 = 0.97\,10^{-3},\ \tau^{2q-1} = 0.015625$		
Time step	Relative error	$\|u_{exact} - u_{IRK}\|_\infty$
1	$0.45493\ 10^{-2}$	$0.79309\ 10^{-2}$
5	$0.24133\ 10^{-2}$	$0.61556\ 10^{-3}$
10	$0.35750\ 10^{-2}$	$0.65786\ 10^{-2}$
15	$0.68736\ 10^{-2}$	$0.17060\ 10^{-2}$
20	$0.14379\ 10^{-2}$	$0.12306\ 10^{-2}$
25	$0.43298\ 10^{-2}$	$0.55986\ 10^{-2}$

Table 5.1

Case (E2): Problem size 4226*, average number of GCR iterations* 4

(E2) At each time step, solve $\mathcal{A}$ iteratively by the Generalized Conjugate Residual (GCR) method ([25]), preconditioned by $\mathcal{P}$ with relative stopping tolerance $\varepsilon_{gcr} = 10^{-12}$. The diagonal blocks in $\mathcal{P}_d$ are solved via a direct method.

(E3) Systems with $\mathcal{A}$ are solved as in (E2), however, the diagonal blocks $\lambda_k M - \tau K$, $k = 1, 2, \cdots, q$ are solved via AGMG ([26]). To save time in constructing the corresponding AGMG preconditioner for the different eigenvalues of L, all blocks are preconditioned by AGMG, constructed for $\max\{\lambda_k\} M - \tau K$.

In all three cases the Euclidean norm $\|u_{exact} - u_{IRK}\|_2$ and the infinity norm $\|u_{exact} - u_{IRK}\|_\infty$ are found the same. Therefore we present only some results for Case (E2), Tables 5.1, 5.2, 5.3 and for Case (E3) - Table 5.4. We see that the iterative method does not lead to loss of accuracy, compared to the direct solution with $\mathcal{A}$. This is achieved by setting the outer stopping tolerance to be nearly the machine accuracy. The tolerance is met after a very few iterations, thanks to the efficient preconditioner $\mathcal{P}$, allowing also for full parallelism between the stages. In addition, the diagonal blocks $\lambda_k M - \tau K$, which are also of large dimensions, can be efficiently solved by some optimal or nearly optimal iterative methods of algebraic multigrid or multilevel type, such as AGMG, preserving both the outer convergence and the accuracy of the result. Last but not least, all diagonal blocks can be preconditioned by one and the same preconditioner, based on the matrix, corresponding to the largest eigenvalue of Λ. As it is further explained in the concluding remarks, this leads to huge savings in the computational labor and elapsed time.

6. Concluding remarks. It follows that using a higher order IRK method can lead to huge savings in computer labor and time. Compare, e.g., the backward Euler method (i.e. $q = 1$) with time-step τ. To balance its local discretization error $O(\tau^2)$, we use an IRK method, of order $q = 6$, that is, with timestep τ_1 chosen so that $\tau_1^{2q} = O(\tau^2)$, say, $\tau_1 = O(\tau^{1/q})$. If say $\tau^2 = 10^{-6}$ we can then choose $\tau_1 = 10^{-1/2}$. Hence, IRK needs T/τ_1, integration steps instead of $T/\tau = 10^3 T$. To solve the outer block system to a given tolerance, about six iterations are needed. But since one can use parallel computations for the IRK method when solving the q block matrix systems, we save computational labor and elapsed computer time with a factor $10^3/6 \cdot 10^{-1/2} \approx 50$. Requiring higher accuracy, this factor will increase even further.

Acknowledgments The work of the first author is supported by The Ministry of Education, Youth and Sports of the Czech Republic from the National Programme of Sustainability (NPU II), project “IT4Innovations excellence in science - LQ1602” and the work of the second authors - by Grant 2017-03749 of the Swedish Research

$q = 3$		
$\tau = 2^{-2},\ h = 2^{-7},\ h^2 = 0.61035\,10^{-4},\ \tau^{2q-1} = 0.98\,10^{-4}$		
Time step	Relative error	$\Vert u_{exact} - u_{IRK}\Vert_\infty$
1	$0.15369\ 10^{-3}$	$0.27489\ 10^{-3}$
5	$0.48073\ 10^{-3}$	$0.13502\ 10^{-3}$
10	$0.13282\ 10^{-3}$	$0.25110\ 10^{-3}$
15	$0.68090\ 10^{-3}$	$0.16767\ 10^{-3}$
20	$0.16816\ 10^{-3}$	$0.13848\ 10^{-3}$
23	$0.68090\ 10^{-3}$	$0.15171\ 10^{-3}$
24	$0.25088\ 10^{-3}$	$0.19298\ 10^{-3}$
25	$0.16305\ 10^{-3}$	$0.21555\ 10^{-3}$

Table 5.2

Case (E2): Problem size 99075*, average number of GCR iterations* 5

$q = 5$		
$\tau = 2^{-1},\ h = 2^{-7},\ h^2 = 0.61035\,10-4,\ \tau^{2q-1} = 0.195\,10^{-2}$		
Time step	Relative error	$\Vert u_{exact} - u_{IRK}\Vert_\infty$
1	$0.21016\ 10^{-3}$	$0.42859\ 10^{-3}$
2	$0.22418\ 10^{-3}$	$0.22251\ 10^{-3}$
4	$0.19455\ 10^{-3}$	$0.18469\ 10^{-3}$
5	$0.21377\ 10^{-3}$	$0.39435\ 10^{-3}$
8	$0.19456\ 10^{-3}$	$0.16711\ 10^{-3}$
9	$0.21377\ 10^{-3}$	$0.35682\ 10^{-3}$
16	$0.19456\ 10^{-3}$	$0.13682\ 10^{-3}$
17	$0.21377\ 10^{-3}$	$0.29214\ 10^{-3}$
20	$0.19456\ 10^{-3}$	$0.12380\ 10^{-3}$
21	$0.21377\ 10^{-3}$	$0.26434\ 10^{-3}$
24	$0.19456\ 10^{-3}$	$0.11202\ 10^{-3}$
25	$0.21377\ 10^{-3}$	$0.23918\ 10^{-3}$

Table 5.3

Case (E2): Problem size 165125*, average number of GCR iterations* 5

				Stopping tolerance (inner solver)			
q	τ	h	Size of K	10^{-6}		10^{-3}	
				Outer it.	Inner it.	Outer it.	Inner it.
2	2^{-2}	2^{-5}	2113	6	13	7	7
3	2^{-2}	2^{-7}	33025	6	16	8	10
5	2^{-1}	2^{-7}	33025	7	16	10	10
6	2^{-1}	2^{-7}	33025	8	16	11	10

Table 5.4

Case (E3): average iteration counts of the outer (GCR) iterations per time step and the inner (AGMG) iterations per GCR iteration

Council VR, *Mathematics and numerics in PDE-constrained optimization problems with state and control constraints*, 2018-2022.

REFERENCES

[1] J. Crank and P. A. Nicolson, *Practical Method for Numerical Evaluation of Partial Differential Equations of the Heat Conduction Type*, Proc. Camb. Phil. Soc., 1 (1947), pp. 50–67.

[2] G. Dahlquist, *A special stability problem for linear multistep methods*, BIT, 3 (1963), pp. 27–43.

[3] O. Axelsson, *A class of A-stable methods*, Nordisk Tidskr. Informationsbehandling (BIT), 9 (1969), pp. 185–199.

[4] B. L. Ehle, *High order A-stable methods for the numerical solution of D.E.s*, BIT, 8 (1968), pp. 276–278.

[5] J. C. Butcher, *Implicit Runge-Kutta processes*, Math. Comput., 18 (1964), pp. 50–64.

[6] O. Axelsson, *Global integration of differential equations through Lobatto quadrature*, Nordisk Tidskr. Informationsbehandling, 4 (1964), pp. 69–86.

[7] B. L. Ehle, *A-stable methods and Padé approximations to the exponential*, SIAM J. Math. Anal., 4 (1973), pp. 671–680.

[8] J. D. Lambert, *Numerical Methods for Ordinary Differential Systems*, Wiley, New York, 1992.

[9] L. Petzold, *Order results for implicit Runge-Kutta methods, applied to differential-algebraic systems*, SIAM J Numer Anal., 23 (1986), pp. 837–852.

[10] O. Axelsson, R. Blaheta, and R. Kohut, *Preconditioning methods for high-order strongly stable time integration methods with an application for a DAE problem*, Numer. Linear Algebra Appl., 22 (2015), pp. 930–949.

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

[12] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, Springer-Verlag, 1991.

[13] O. Axelsson, *On the efficiency of a class of A-stable methods*, Nordisk Tidskr. Informationsbehandling (BIT), 14 (1974), pp. 279–287.

[14] P. J. van der Houwen and B. P. Sommeijer, *Analysis of parallel diagonally implicit iteration of Runge-Kutta methods*, Parallel methods for ordinary differential equations (Grado, 1991). *Appl. Numer. Math.* 11 (1993), pp. 169–188.

[15] J. C. Butcher, *Diagonally-implicit multi-stage integration methods*, Appl. Numer. Math, 11 (1993), pp. 347–363.

[16] Z. Zlatev, *Modified diagonally implicit Runge-Kutta methods*, SIAM J. Sci. Statist. Comput., 2 (1981), pp. 321–334.

[17] T. J. R. Hughes and G. M. Hulbert, *Space-time finite element methods for elastodynamics: Formulations and error estimates*, Comput. Methods Appl. Math., 66 (1988), pp. 339–363.

[18] K. Eriksson and C. Johnson, *Adaptive finite element methods for parabolic problems I: A linear model problem*, SIAM J. Numer. Anal., 28 (1991), pp. 43–77.

[19] O. Steinbach and H. Yang, *Comparison of algebraic multigrid methods for an adaptive space-time finite-element discretization of the heat equation in 3D and 4D*, Numer. Linear Algebra Appl., 25 (2018), e2143.

[20] J. Shohat, *On mechanical quadratures, in particular, with positive coefficients*, Trans. Amer. Math. Soc., 42 (1937), pp. 461–496.

[21] O. Axelsson, D. Lukáš, *Preconditioning methods for eddy-current optimally controlled time-harmonic electromagnetic problems*, J. Numer. Math., 27 (2019), pp. 1–21.

[22] H. Chen, *Kronecker product splitting preconditioners for implicit Runge-Kutta discretizations of viscous wave equations*, Appl. Math. Modelling, 40 (2016), pp. 4429–4440.

[23] E. Hairer, C. Lubich and M. Roche, *Error of Runge-Kutta methods for stiff problems studied via differential algebraic equations*, BIT, 28 (1988), pp. 678–700.

[24] O. Axelsson, R. Blaheta and T. Luber, *Preconditioners for mixed FEM solution of stationary and nonstationary porous media flow problems*, LSSC 2015, I. Lirkov et al (Eds.), LNCS 9374 (2015), pp. 3–14.

[25] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, 2003.

[26] Y. Notay, *An aggregation-based algebraic multigrid method*, ETNA, 37 (2010), pp. 123–146.