Received: 30 May 2022 | Revised: 29 August 2023 | Accepted: 4 September 2023

DOI: 10.1002/nla.2532

RESEARCH ARTICLE

WILEY

Stage-parallel preconditioners for implicit Runge–Kutta methods of arbitrarily high order, linear problems

Owe Axelsson[1] | **Ivo Dravins**[2] | **Maya Neytcheva**[1]

[1]Department of Information Technology, Uppsala University, Uppsala, Sweden

[2]Faculty of Mathematics, Ruhr University Bochum, Bochum, Germany

Correspondence
Ivo Dravins, Faculty of Mathematics, Ruhr University Bochum, Bochum, Germany.
Email: ivo.dravins@rub.de

Funding information
Vetenskapsrådet, Grant/Award Numbers: VR-2017-03749, 2018-05973; Swedish National Infrastructure for Computing, Grant/Award Number: 2021/22-633

Abstract

Fully implicit Runge–Kutta methods offer the possibility to use high order accurate time discretization to match space discretization accuracy, an issue of significant importance for many large scale problems of current interest, where we may have fine space resolution with many millions of spatial degrees of freedom and long time intervals. In this work, we consider strongly A-stable implicit Runge–Kutta methods of arbitrary order of accuracy, based on Radau quadratures. For the arising large algebraic systems we introduce efficient preconditioners, that (1) use only real arithmetic, (2) demonstrate robustness with respect to problem and discretization parameters, and (3) allow for fully stage-parallel solution. The preconditioners are based on the observation that the lower-triangular part of the coefficient matrices in the Butcher tableau has larger in magnitude values, compared to the corresponding strictly upper-triangular part. We analyze the spectrum of the corresponding preconditioned systems and illustrate their performance with numerical experiments. Even though the observation has been made some time ago, its impact on constructing stage-parallel preconditioners has not yet been done and its systematic study constitutes the novelty of this article.

KEYWORDS

fully stage-parallel preconditioning, implicit Runge–Kutta methods, parallelization, Radau quadrature

1 | INTRODUCTION

The availability of substantial computational power and high performance computing resources has enabled high resolution, both in space and time, when performing numerical simulations of numerous problems of interest, modeling processes in physics, computational biology, financial and social processes, to name a few application areas.

Combining the requirements to have a fine, sometimes very fine, space resolution, high enough time resolution to balance the discretization error in space and time, fast and robust numerical simulations tilt the scales in favor of space discretization methods that easily handle complex geometries and adaptivity, together with higher order implicit time discretization methods. In this study, we assume that the space discretization is done applying suitable finite element methods (FEM) and focus on a particular class of implicit time discretization schemes, namely, the implicit

Runge–Kutta (IRK) methods, based on Radau (also referred to as Radau IIA) quadratures. To be specific, consider a system of evolutionary equations of the form

$$M\frac{\partial \boldsymbol{u}(t)}{\partial t} + K\boldsymbol{u}(t) = \boldsymbol{f}(t), \tag{1}$$

arising after semidiscretization in space of some non-stationary linear partial differential equation of the type $\frac{\partial u}{\partial t} - \varepsilon\Delta u + \boldsymbol{b}\cdot\nabla u = f(t)$, equipped with appropriate initial and boundary conditions. In (1), M is a mass matrix, K is a stiffness matrix and $\boldsymbol{u}(t), \boldsymbol{f}(t)$ are vectors.

As the name suggests, evolution equations describe the time evolution of a physical system starting from given initial data.

In many real-life problems, where we need to resolve fine features of the solution during the time evolution, Such problems are, for instance, convection-diffusion problems where various types of layers are developing in time. In this study, we exclude from consideration explicit time integrators. When we use a stable implicit time-integrator such as the backward Euler, the trapezoidal method or the Crank–Nicolson method (cf. Reference 1), the time-step must still be small to guarantee a sufficiently small time-integration error that balances a small space discretization error. Therefore, there are strong reasons to apply higher order time-integration methods, which can enable the usage of much fewer time-steps and combine small total discretization error with fast integration in time.

As is also well known, see for example, Reference 2, classical multistep methods cannot have a higher order than two, otherwise they are not stable for all eigenvalues of the evolution operator $M^{-1}K$, that is, they are not A-stable. This can be a severe limitation because in many problems, such as network problems and time-harmonic Maxwell's equations there can appear rapidly changing oscillations, leading to the appearance of eigenvalues, widespread in the whole right half complex plane. On the other hand, in Reference 3 see also Reference 4, it is proven that there exist implicit Runge–Kutta methods of an arbitrary high order that are A-stable.

Implicit Runge–Kutta methods, known also as collocation methods, are first presented in Reference 5, giving the methods the name "IRK." Their general form to solve a problem of the form $u' = F(t, u(t))$ reads:

$$u^{(k+1)} = u^{(k)} + \tau\sum_{i=1}^{q} b_i F\left(t_k + c_i\tau, v_i^{(k)}\right),$$

and the so-called *stage* variables are computed as $v_i^{(k)} = u^{(k)} + \tau\sum_{j=1}^{q} a_{ij}F(t_k + c_j\tau, v_j^{(k)}), i = 1, \ldots, q$. The method ingredients are the number of stages q, the Runge–Kutta matrix $A_q = [a_{ij}]$, the weights b_i that depend on the particular integration method, and the stage nodes c_i, which are the zeros of the shifted Legendre polynomials of degree q, defined in the interval $[0, 1]$ and normalized at 0, satisfying $\sum_{i=1}^{q} c_i = 1$. Further, τ is the time step, $u^{(k)}$ and $u^{(k+1)}$ are the numerically computed solutions at time-steps k and $k + 1$.

Independently, such methods are presented in Reference 6 and it is shown that due to their high order of approximation and stability properties, they could be considered as global integration methods, that is, it could suffice to use just one or very few time-steps, that is, 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 References 7 and 3.

There are several versions of IRK methods.[8-11] The most familiar methods are based on inner interval integration points being the zeros of some particular polynomials, namely, the Gauss integration method (G), the Radau or Gauss-Radau integration method (R) and the Lobatto or Gauss-Lobatto integration method (L).

The approximation order (p) at the endpoints of each time-interval of methods (G), (R), and (L) is correspondingly $p = 2q$, $2q - 1$, and $2q - 2$. The approximation order at the interior integration points, however, is only of order q, compare, for example, Reference 5. We stress, that IRK are particularly important for large time intervals where we are somewhat less interested in the behavior of the solution in intermediate time instances but rather mainly of the solution of the underlying evolution systems at the final time in the considered time interval.

All methods (G), (R), (L) are A-stable, compare, for example, Reference 12. However, methods (G) and (L) are not *strongly* A-stable because for them the absolute value of the stability function converges to unity, implying, in the presence of large eigenvalues of K or the Jacobian matrix, at least a linear growth of rounding errors with increasing number of repeated time steps. In this work, we advocate the Radau method which is the only *strongly* A-stable (also called *strongly* L-stable) method, for a definition, see for instance.[6,9,13] Furthermore, the Radau method (referred to as stiffly accurate,

compare, e.g., Reference 9) does not suffer from order reduction, that can occur for instance in the solution of systems of differential-algebraic equations, see, for example, References 14 and 15.

The high accuracy of the IRK methods makes them very attractive, in particular, in the case when the underlying problem requires a very fine space resolution and the time interval to integrate over is large. The drawback with IRK is that in each timestep we have to solve the arising large algebraic system of order qn, where q is the number of stages of the method, that is, equals the degree and number of zeros of the Legendre or the combination of Legendre polynomials and n is the size of M and K. For general IRK methods all stages are coupled and cannot be straightforwardly decomposed in some easier to handle form. The solution of this system can be costly and somewhat involved, which has been the major reason why IRK methods are more rarely used. In practice, the system must be solved by some preconditioned iterative method, preferably, implemented efficiently in a parallel computer environment.

We mention for completeness, that in order to avoid the complications when employing implicit time-integration methods, there have been efforts in applying Richardson extrapolation and similar techniques in combination with explicit Runge–Kutta methods. The extrapolation still enables extension of the stability region, see, for example, Reference 16 and 17. However, in addition to be forced to choose time-steps to make a stable recursion, the methods do not have the high order of discretization error, inherent in the IRK methods.

For an earlier discussion of solution techniques for IRK methods, see References 18-20, also References 21 and 22, 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, force the use of smaller time-steps. Since some time an alternative approach to solve time-dependent partial differential equations have been used, see, for example, References 23-25, based on combined time-space finite elements. This means that a 2D space partial differential operator is solved in a 3D space-time domain and a 3D space partial differential operator is solved in a 4D space-time finite element mesh. Clearly this complicates the implementation of the method, but such methods enable the use of adaptive mesh resolution methods in both time and space.

As stated above, in this article, we consider the solution of time-dependent partial differential equations and, to motivate the need of large timesteps, assume that the time interval is large.

The major topic of this study is the construction of preconditioning methods for full IRK. The preconditioners considered here differ from other similar preconditioners, being based on a simple observation from Reference 6, which requires only real computations, enables very numerically efficient preconditioning techniques, which are also fully parallel with respect to the stages. The novelty in this study is that we formulate, analyze, implement the suggested preconditioning methods and show their robustness and good parallelization properties on problems of varying difficulty. The construction of the preconditioners is based on the inherent properties of the Runge–Kutta matrices and not on purely algebraic manipulations with the matrices as in other studies. To our knowledge, this is the first real-arithmetic, discretization-robust, stage-parallel fully implicit Runge–Kutta preconditioning approach.

In this study, we implement and show the performance of the methods on large scale distributed memory high performance computer platform for the stage-serial case. Stage-parallel implementations using a memory efficient matrix-free FEM framework—including comparisons to the stage-serial case and implementation details for example, necessary communication patterns, virtual typologies and so forth warrant a separate study, available in Reference 26. (For completeness we state that the idea is first illustrated in Reference 27 without rigorous theoretical background and parallel performance tests.)

The article is structured as follows. In Section 2, we present the Radau type of IRK methods. Section 3 summarizes earlier approaches to solve the large IRK matrices efficiently and in parallel. The preconditioners, suggested in this article are stated in Section 4 and analyzed in Section 5. Section 6 discusses implementation issues and Section 7 contains illustrative numerical experiments. Conclusions and outlook are found in Section 8.

For more details we refer to the technical report in Reference 28.

2 | IMPLICIT RUNGE–KUTTA METHODS

Time-integration methods are repeated on each time-step, whereby the end solution of the previous step is used as the initial solution for the next time interval. Therefore, we do not use overlapping time-steps but instead advocate to utilize

high order methods and very long time steps. To describe the method it suffices to consider just a single time-integration interval, $(0, \tau)$, $\tau > 0$, without any overlap.

In the subsequent derivations I_m denotes the identity matrix of order m. We utilize also the following tensor algebra identity with $\otimes$ being the matrix tensor product, known as well as Kronecker product,

$$(a \otimes b)(c \otimes d) = (ac) \otimes (bd). \tag{2}$$

2.1 | Discrete tensor product matrix forms

To give an instance, consider the differential equation (1), that is,

$$M\frac{d\boldsymbol{u}(t)}{dt} + K\boldsymbol{u}(t) = \boldsymbol{f}(t), \quad 0 < t \le \tau, \quad \boldsymbol{u}(0) = \boldsymbol{u}_0.$$

This can be written as

$$M\boldsymbol{v}(t) + K\int_0^t \boldsymbol{v}(s)ds = \boldsymbol{f}(t) - K\boldsymbol{u}_0, \tag{3}$$

where $\boldsymbol{v}(t) = \frac{d\boldsymbol{u}(t)}{dt}$, that is, $\boldsymbol{u}(t) = \int_0^t \boldsymbol{v}(s)ds + \boldsymbol{u}_0$.

Following Reference 3, to solve (3) we approximate the integral term as $\sum_{i=0}^{k} a_{ik}\boldsymbol{v}(c_i)$, by using Radau quadrature. Let ℓ_q be the shifted Legendre polynomial of degree q, normalized at 0. We integrate from 0 to τc_i, $i = 1, \ldots, q$, where c_i are the zeros of $\ell_q(t) - \ell_{q-1}(t)$, $0 < t \le 1$, use the Lagrange interpolation polynomials for the points c_i, $l_k(z) = \prod_{i=1,i\neq k}^{q}(z - c_i)\Big/\prod_{i=1,i\neq k}^{q}(c_k - c_i)$ to represent $\boldsymbol{v}(z) \approx \sum_{k=1}^{q}\boldsymbol{v}_k l_k(z)$, we obtain that $a_{ik} = \int_0^{c_i} l_k(z)dz, \quad i, k = 1, \ldots, q$. As shown in for example, References 29 and 30, the matrix $A_q = [a_{ik}]_{i,k=1}^{q}$ can be presented in the form of a product of four matrices,

$$A_q = CVRV^{-1},$$

where $C = \mathrm{diag}\{c_1, c_2, \ldots, c_q\}$, $R = \mathrm{diag}\{1, 1/2, \ldots, 1/q\}$ and V is the Vandermonde matrix, generated by c_i, that is,

$$V = \begin{bmatrix} 1 & c_1 & \cdots & c_1^{q-1} \\ \cdots & \cdots & \cdots & \cdots \\ 1 & c_q & \cdots & c_q^{q-1} \end{bmatrix}.$$

Since the zeros $\{c_i\}$ of the Legendre polynomials are distinct, V is nonsingular, thus, invertible. Clearly the IRK quadrature matrix A_q is nonsingular too.

Having computed A_q, by use of the numerical integration $\int_0^{c_i}\widetilde{\boldsymbol{v}}(s)\,ds = \sum_{k=1}^{q} a_{ik}\widetilde{\boldsymbol{v}}(c_k)$, $i = 1, 2, \ldots, q$, and using tensor product representation, we obtain the algebraic form of (3),

$$(I_q \otimes M + \tau A_q \otimes K)\boldsymbol{v} = \boldsymbol{f} - (I_q \otimes K)(\boldsymbol{e}_q \otimes \boldsymbol{u}_0), \tag{4}$$

where $\boldsymbol{f} = [f_i]_{i=1}^{q}$, $f_i = f(\tau c_i)$ and $\boldsymbol{e}_q$ is a vector of length q with all components 1. The matrices M and K are real of size $n \times n$. The block vector $\boldsymbol{v}$ of length qn has block components $\boldsymbol{v}_i, i = 1, \ldots, q$ and each $\boldsymbol{v}_i$ is of length n. Multiplying (4) from the left by $A_q^{-1} \otimes I_n$ and utilizing (2) we obtain the alternatively transformed form of (4),

$$\left(A_q^{-1} \otimes M + \tau I_q \otimes K\right)\boldsymbol{v} = \left(A_q^{-1} \otimes I_n\right)\boldsymbol{f} - \left(A_q^{-1} \otimes K\right)(\boldsymbol{e}_q \otimes \boldsymbol{u}_0). \tag{5}$$

This transformation is suggested in Reference 31, see also Reference 29. The systems (4) and (5) involve the $qn \times qn$ block matrices, respectively,

$$\mathcal{A}_1 = I_q \otimes M + \tau A_q \otimes K \quad \text{and} \quad \mathcal{A}_2 = A_q^{-1} \otimes M + \tau I_q \otimes K. \tag{6}$$

The IRK framework is conveniently formulated via the so-called Butcher tableau,[32] that is, $\left[\begin{array}{c|c} \boldsymbol{c} & A_q \\ \hline & \boldsymbol{b} \end{array}\right]$.We note here, that for the Radau quadrature method $c_q = 1$, then $b_i = a_{qi}$, $i = 1, 2, \dots, q$. For further comments, see Reference 29.

Explicit forms of the matrices A_q and A_q^{-1} for various values of q are given in Reference 28 Appendix A.

3 | A SUMMARY OF SOME EARLIER PRESENTED STAGE-PARALLEL PRECONDITIONING METHODS

Clearly, for large scale algebraic problems, to solve systems with the matrices $\mathcal{A}_1$ or $\mathcal{A}_2$ in (6), using an exact block matrix factorization method in a straightforward manner is infeasible as it would lead to full matrices and would be clearly too expensive in computer time and memory demands. The way to reduce the computer resource demands is, instead of direct solution methods, to use a preconditioned iterative solution method, such as the generalized conjugate gradient method (GCG)[33] or the generalized minimum residual (GMRES) method,[11] entailing the task to construct an efficient preconditioner.

Over many years, there have been various attempts to construct parallel solution methods for IRK problems, either by approximating the method by some lower order method on simpler form, by constructing a parallel preconditioner or by extending the method to a multistage method on several time steps. We list here some of these methods.

There are three general types of approaches that achieve some parallelism—across the method (across stages), across the problem, and across the time-steps.

For a more general discussion of parallelization in Runge–Kutta methods, we refer to Reference 34.

An early attempt to achieve parallelism across the method is to apply diagonally-implicit Runge–Kutta methods, also called DIRK methods, see, for example, References 35 and 19, also Reference 22. However, as already mentioned, the methods have lower order of approximation (for a given number of stages), and thus require smaller and, therefore, relatively many time steps. Another possibility is in using a diagonal matrix as a preconditioner to the IRK method. An early experience is found in Reference 36. There, the linear systems are decoupled into subsystems, which means that the cost of the full LU factorization of the global matrix is reduced to q factorizations of the diagonal blocks of size n.

Examples of approaches to achieve parallelism across the problem can be found in Reference 37, see also the references therein, exploring the possibility to solve the problem in subintervals in parallel. Further, as pointed out in Reference 36, the construction of the preconditioner can be based on the Vandermonde matrix representation of the quadrature matrix.

The parareal approach, for example, in Reference 38 is to seek parallelism across time.

3.1 | Using decompositions of A_q or A_q^{-1}

One strategy to construct a stage-parallel preconditioner is based on the spectral decomposition of the quadrature matrix A_q,

$$T_1^{-1} A_q T_1 = \Upsilon_1 = \mathrm{diag}\{\upsilon_1^{(1)}, \upsilon_2^{(1)}, \dots, \upsilon_q^{(1)}\}.$$

The matrix $\mathcal{A}_1$ (6) can then be transformed into a factorized form, namely,

$$\mathcal{A}_1 = I_q \otimes M + \tau A_q \otimes K = \left(T_1^{-1} \otimes I_n\right) \mathcal{A}_{1,ii} (T_1 \otimes I_n), \quad \mathcal{A}_{1,ii} = I_q \otimes M + \tau \Upsilon_1 \otimes K. \tag{7}$$

The second option is to consider $T_2^{-1} A_q^{-1} T_2 = \Upsilon_2 = \mathrm{diag}\left\{\upsilon_1^{(2)}, \upsilon_2^{(2)}, \dots, \upsilon_q^{(2)}\right\}$ and obtain

$$\mathcal{A}_2 = A_q^{-1} \otimes M + \tau I_q \otimes K = \left(T_2^{-1} \otimes I_n\right) \mathcal{A}_{2,ii} \left(T_2 \otimes I_n\right), \quad \mathcal{A}_{2,ii} = \Upsilon_2 \otimes M + \tau I_q \otimes K. \tag{8}$$

We see that $\mathcal{A}_{1,ii}$ and $\mathcal{A}_{2,ii}$ are block-diagonal matrices that decouple the stages. However, some of the eigenvalues of A_q or A_q^{-1} are complex, which entails the usage of complex arithmetic and increases the computational cost. We refer to Reference 39, where a Jordan normal form based approach is suggested, which, if A_q is diagonalizable, allows the transformation to a block diagonal system, enabling stage parallelism on the cost of, in general, solving complex systems for the complex-conjugate pairs of eigenvalues in real arithmetic. One can exploit the complex-conjugate property in the

solver but some standard techniques for solving complex systems limit the achievable performance—see Reference 26 for a detailed comparison.

In Reference 31, ILU-based stage-coupled and stage-parallel preconditioners are presented. There, the stage-parallel preconditioner is the block-diagonal part of the block-LU factorization of the system matrix, where the blocks are ILU-factorized themselves. The experiments indicate that the stage-parallel (referred to as stage-uncoupled) preconditioner is numerically less efficient than that of the stage-coupled version.

A comparison study of a series of LDU-based preconditioning approaches, applied on the non-transformed Equation (4) is recently presented in Reference 40. The authors test various combinations of the LDU factorization of A_q. The preconditioners are implemented in a block-Jacobi and Gauss–Seidel fashion. The numerical results, presented there, indicate that the $\mathcal{P}_{LD}$ option is most robust in terms of small condition number. The approach does not utilize any particular property of the Runge–Kutta matrix and does not provide estimates of the spectrum or the condition number of the arising preconditioned matrices.

Another recent work on preconditioning IRK, related to the considered framework, is found in Reference 41. The preconditioner is based on a form of the system matrix, similar to that of $\mathcal{A}_2$, and on a closed form of the inverse of the matrix $A_q^{-1} \otimes I_n + \tau I_q \otimes M^{-1}K$. The method presented there exploits the complex-conjugate property of the eigenvalues. It is inherently stage-serial as it requires the sequential solution of stage(-pairs).

3.2 | Transforming A_q to avoid complex arithmetic

To construct a preconditioner that requires only real arithmetic, it is suggested to use some transformation of A_q, for instance,

$$A_q = CVRV^{-1} = B^{-1}W^{-T}XW^{-1}, \tag{9}$$

where the entries of W are computed as $w_{ij} = \ell_{j-1}(c_i)$ with $\ell_i(x)$ being the shifted Legendre polynomial of degree i and $B = \mathrm{diag}(b_1, b_2, \ldots, b_q)$. The matrix X is tridiagonal. The above W-transformation is first published in Reference 8 and also advocated in Reference 42. In this way the complex arithmetic is avoided and the decoupling cost for the parallel implementation of this approach corresponds to that for the implicit Euler method. In Reference 42, the factorization (9) is used to obtain a preconditioner of block-tridiagonal form, which is then LU-factorized and simplified using a method parameter to be determined.

3.3 | Other approaches

In Reference 43, standard preconditioners for low-order time discretizations are used to construct order-optimal diagonal block Jacobi preconditioners for high order discretizations. The convergence properties of the methods are improved in Reference 44 by employing block Gauss–Seidel techniques.

In Reference 45, optimal complex and real Schur-based preconditioners are compared, together with a block Jordan-form preconditioner and a near optimal singly diagonal approximate block real Schur decomposition is derived. The latter in particular has memory requirements and setup cost comparable to singly DIRK methods.

An example of a parallel across the problem method is the waveform relaxation technique, see for instance, Reference 46. A drawback to mention here is that for stiff problems the convergence of these methods to the exact solution can be very slow. A similar technique is to use a time-harmonic expansion of the solution, which is a natural approach for problems with alternating source functions, such as in time-harmonic electromagnetic problems, compare References 47 and 48 and the references therein. In such an approach the problem decouples into independent subproblems per angular frequency and in this way there is no need to use any time-integration method at all. Another approach to use Fourier expansions is to extend the interval $[0, \tau]$ to $[0, 2\tau]$ and use the backward form of the IRK method on the symmetrical interval $[\tau, 2\tau]$. Here the solution can be expanded in $e^{i\omega_k t}$ terms, which, again leads to uncoupled problems for each frequency ω_k and can be solved fully in parallel, compare Reference 48.

For parabolic problems of the form

$$\boldsymbol{u}_t + K\boldsymbol{u} = \boldsymbol{0}, \ \boldsymbol{u}(0) = \boldsymbol{u}_0,$$

one can use the Laplace transform with parameter s. We then have

$$s\widehat{\boldsymbol{u}} + K\widehat{\boldsymbol{u}} = \boldsymbol{u}_0, \text{ which implies } \widehat{\boldsymbol{u}} = (sI + K)^{-1}\boldsymbol{u}_0.$$

The solution $\boldsymbol{u}(t)$ can be computed via the inverse Laplace transform $\boldsymbol{u}(t) = \frac{1}{2\pi i}\int_{\Gamma} e^{st}\widehat{\boldsymbol{u}}(s)\, ds$, where Γ is a contour in the right half plane, that is, contains no eigenvalues of $sI + A$. Here the integral is approximated with a quadrature formula with nodes s_j. Then one only needs to compute $\widehat{\boldsymbol{u}}(s)$ at all s_j, which can be done fully in parallel. For details, see References 49 and 50.

Another possible parallelization method is the boundary value technique, that has appeared in Reference 51. Here the whole time interval is solved by computing the equation as arising in the implicit trapezoidal method with a backward midpoint rule at the endpoint. The whole system then becomes block tridiagonal and can be solved in various ways by common methods for two-point boundary value problems.

Since long, multigrid methods have successfully been used to solve space discretized problems and also have been shown to be efficiently parallelized. Parallel-in-time, that is, parallelizable across time methods have also been developed, such as multiple-shooting techniques, compare, for example, Reference 52. The parareal framework has further renewed the interest in such methods, see References 38 and 53. We claim that if one can solve the full IRK method in a stage-parallel manner in each time step, there would less need for such multiple timestep methods. However, these can be useful to obtain solution with higher accuracy also in interior integration points. A GPU implementation of the fully implicit IRK method is found in Reference 54, applied to a system of ODEs. There, the matrices to be solved have the same structure as in (7) and the preconditioner has a tridiagonal structure.

Related to IRK methods, however out of the context of parallel implementation, preconditioners for quadratic matrix polynomials are analyzed in Reference 55.

In Reference 56, a monolithic multigrid for IRK methods, applied to incompressible flows is studied. The coupled systems are solved by the designed multigrid, choosing all computational ingredients of the solver to be parallelizable, however, not aiming to achieve parallelization across the stages.

4 | STAGE-PARALLEL PRECONDITIONING METHODS FOR THE RADAU TENSOR PRODUCT SYSTEM

From now on we consider IRK methods based on Radau quadratures (Radau IIA). The derivations apply to other IRK methods for which the Butcher matrices A possess the same properties as those for Radau IIA, discussed in Reference 6.

It is straightforwardly observed that the first term in the matrix $\mathcal{A}_1$, respectively, the second term of the matrix $\mathcal{A}_2$ in (6) involve a block-diagonal matrix and a plausible aim is to get a simpler form of the other matrix term as well. The approach to use the spectral decomposition of A_q achieves that, however, on the cost of using complex arithmetic.

In this study, we retain the framework, however, not using the spectral decomposition of A_q. As shown in Reference 6 (p. 75), for IRK methods based on Radau quadratures, A_q has a dominating lower-triangular part in the sense that it has larger in magnitude entries than the strictly upper triangular part. This property holds also for A_q^{-1} as well as the corresponding exact LU- factorizations of A_q ($A_q = L_q^{(1)}U_q^{(1)}$) and A_q^{-1} ($A_q^{-1} = L_q^{(2)}U_q^{(2)}$). Therefore, a possibility to precondition $\mathcal{A}_1$ and $\mathcal{A}_2$ is by the matrices

$$\mathcal{P}_1 = I_q \otimes M + \tau L_q^{(1)} \otimes K \quad \text{and} \quad \mathcal{P}_2 = L_q^{(2)} \otimes M + \tau I_q \otimes K, \tag{10}$$

respectively, where $L_q^{(1)}$ and $L_q^{(2)}$ are the lower-triangular factors in the exact LU-factorizations of A_q, respectively, of A_q^{-1}. This leads to achieving a block lower-triangular form of $\mathcal{P}_1$ and $\mathcal{P}_2$, however, in this way the solution with them involves successively solving systems with the block-diagonal part $(I_q \otimes M + \tau(L_q^{(1)} \otimes K))_{ii}$ of $\mathcal{P}_1$ and correspondingly for $\mathcal{P}_2$, and does not allow for any parallelism between the stages of the method.

Clearly, a preconditioner based on directly using the lower-triangular part is mainly sequential and although there exist general techniques to handle solutions with block lower-triangular matrices in parallel, the amount of parallelism to achieve is rather limited and we leave this approach out of consideration.

Next we make a choice between $\mathcal{P}_1$ and $\mathcal{P}_2$. For $\mathcal{P}_1$ the elimination requires solutions with matrices $(I_q \otimes M + \tau(L_q^{(1)} \otimes K))_{ii}$. When K is ill-conditioned then these block matrices are also ill-conditioned and we need a good preconditioner of these matrices too, for instance of algebraic multilevel (AMLI) type or of algebraic multigrid (AMG) type method.[57,58] There exists various possibilities to employ parallel solvers for these inner systems, but we do not discuss this any further.

The same type of successive elimination method can be applied also for (5). Here the block-diagonal matrices equal $\mathcal{A}_{2,ii}$ which may have a much better form for the efficiency of the inner solution than $\mathcal{A}_{1,ii}$. Furthermore,

(i) the off-diagonal blocks arise now from the matrix term $A_q^{-1} \otimes M$, which is in general sparser than $A_q \otimes K$, in particular if M is a lumped mass matrix, hence, we save also in memory demand;
(ii) in addition, any ill-conditioning of K does not harm the off-diagonal blocks.

Since solving (5) instead of (4) can significantly reduce the computational cost, it can be preferable and this is also the approach taken in this article. Thus, we consider only the form (5). To simplify the notations, we drop the subscript 2 in $\mathcal{A}_2$ and in $\mathcal{P}_2$, and the superscript $^{(2)}$ in $L_q^{(2)}$, thus, we deal with $\mathcal{A} = A_q^{-1} \otimes M + \tau I_q \otimes K$. We consider here a particular LU factorization of A_q^{-1}, namely, we let $A_q^{-1} = L_q U_q$, where the factor L_q is real-valued lower-triangular and the upper-triangular factor U_q has a unit diagonal, thus, $U_q = I_q + \widehat{U}_q$ with $\widehat{U}_q$ strictly upper-triangular. To take an example, for $q = 2$, then we have

$$A_2^{-1} = \begin{bmatrix} \frac{3}{2} & 0 \\ -\frac{9}{2} & 4 \end{bmatrix} \begin{bmatrix} 1 & \frac{1}{3} \\ 0 & 1 \end{bmatrix}, \text{ thus, } \widehat{U}_2 = \begin{bmatrix} 0 & \frac{1}{3} \\ 0 & 0 \end{bmatrix},$$

that is, $\|\widehat{U}_2\| = 1/3$. For $q = 3$ and $q = 4$ we have correspondingly, $\|\widehat{U}_3\| = 0.4098$, $\|\widehat{U}_4\| = 0.4779$ and

$$A_3^{-1} = \begin{bmatrix} 3.225 & 1.168 & -0.253 \\ -3.568 & 0.775 & 1.053 \\ 5.532 & -7.532 & 5 \end{bmatrix}, L_3 = \begin{bmatrix} 3.225 & 0 & 0 \\ -3.568 & 2.067 & 0 \\ 5.532 & -9.535 & 9 \end{bmatrix}, U_3 = \begin{bmatrix} 1 & 0.362 & -0.079 \\ 0 & 1 & 0.374 \\ 0 & 0 & 1 \end{bmatrix},$$

$$L_4 = \begin{bmatrix} 5.6441 & 0 & 0 & 0 \\ -5.0492 & 2.9419 & 0 & 0 \\ 3.4925 & -5.1747 & 3.1618 & 0 \\ -6.9235 & 8.9548 & -16.6361 & 16 \end{bmatrix}, \quad U_4 = \begin{bmatrix} 1 & 0.3408 & -0.1038 & 0.0308 \\ 0 & 1 & 0.4183 & -0.0949 \\ 0 & 0 & 1 & 0.3869 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Remark 1. We notice, that the lower-triangular part of A_q^{-1} and L_q itself possess the property that their diagonals have alternating signs. Therefore, they can be seen as some sort of *difference matrices*. There holds also that L_q^{-1} are positive matrices (with all entries positive). The property is observed for all q up to 12.

4.1 | Preconditioner, based on the spectral decomposition of L_q

Consider the matrix in (10) (right) to be used as a preconditioner to $\mathcal{A}$,

$$\mathcal{P}_L = L_q \otimes M + \tau I_q \otimes K. \tag{11}$$

The subscript $'L'$ denotes that this preconditioner uses the complete factor L_q. If applied straightforwardly, solutions of systems with $\mathcal{P}_L$ do not allow parallelism across stages. Similarly to some earlier attempts towards overcoming this and aiming to obtain a fully stage-parallel preconditioner, compare, for example, References 39 and 59, instead of using the spectral decomposition of A_q^{-1} we use that of L_q.

So, we assume that L_q is a good approximation to A_q^{-1} and construct the spectral decomposition $L_q = T_q \Lambda_q T_q^{-1}$. Here, T_q contains the eigenvectors of L_q and Λ_q is the diagonal part of L_q that contains the eigenvalues of L_q, all real. We then

factor $\mathcal{P}_L$ as

$$\begin{aligned}\mathcal{P}_L &= L_q \otimes M + \tau I_q \otimes K = T_q \Lambda_q T_q^{-1} \otimes M + \tau T_q T_q^{-1} \otimes K \\ &= (T_q \otimes I_n)(\Lambda_q \otimes M)(T_q^{-1} \otimes I_n) + \tau (T_q \otimes I_n)(I_q \otimes K)(T_q^{-1} \otimes I_n) \\ &= (T_q \otimes I_n)\big((\Lambda_q \otimes M) + \tau (I_q \otimes K)\big)(T_q^{-1} \otimes I_n) \\ &= (T_q \otimes I_n)\ \mathcal{P}_d\ (T_q^{-1} \otimes I_n),\end{aligned} \tag{12}$$

where $\mathcal{P}_d = \Lambda_q \otimes M + \tau(I_q \otimes K)$ is block-diagonal. Then, the solution of the generic system $\mathcal{P}_L \boldsymbol{x} = \boldsymbol{f}$ is performed in the following steps:

(1) Solve $\mathcal{P}_d \boldsymbol{y} = (T_q^{-1} \otimes I_n)\boldsymbol{f}$,
(2) Recover $\boldsymbol{x} = (T_q \otimes I_n)\boldsymbol{y}$.

The eigenvectors of L_q (the columns of T_q) can be computed by a simple recursion, as shown in Reference 28 Appendix B, or obtained by some available linear algebra software.

4.2 | Preconditioner, based on a diagonal approximation of L_q

Let $\ell = \max\{L_{q,ii}, i = 1, \dots, q\}$ be the maximum eigenvalue of L_q. Consider the matrix

$$\mathcal{P}_D = \ell I_q \otimes M + \tau I_q \otimes K \tag{13}$$

as a preconditioner to $\mathcal{A}$. The consequence of the approach (13) is that all blocks in $\ell\, I_q \otimes M$ are the same and in the computations we do not need to involve the eigenvector matrices T_q. The numerical examples in Reference 27 are made using the latter framework. There, no theoretical analysis is provided and some preliminary experiments in Matlab are included. We analyse the spectral properties of the corresponding preconditioned matrix in Section 5.2.

Remark 2. Clearly, the approach can be applied to A_q, which is tested in Reference 40. The variant $\mathcal{P}_{LD}$ from that article would correspond to our approach when applied to A_q. As mentioned, there it is found empirically that this variant is numerically most efficient, without considering any particular properties of A_q.

5 | SPECTRAL PROPERTIES OF THE PROPOSED PRECONDITIONERS

We analyze the spectral properties of the preconditioned matrices $\mathcal{P}_L^{-1}\mathcal{A}$ and $\mathcal{P}_D^{-1}\mathcal{A}$.

5.1 | Spectral properties of the preconditioned matrix $\mathcal{P}_L^{-1}\mathcal{A}$

We make first the general observation that

$$\begin{aligned}\mathcal{P}_L^{-1}\mathcal{A} &= (L_q \otimes M + \tau I_q \otimes K)^{-1}(A_q^{-1} \otimes M + \tau I_q \otimes K) \\ &= (L_q \otimes M + \tau I_q \otimes K)^{-1}\Big[(L_q \otimes M + \tau I_q \otimes K) + L_q \widehat{U}_q \otimes M\Big] \\ &= I_{qn} + (L_q \otimes M + \tau I_q \otimes K)^{-1}(L_q \widehat{U}_q \otimes M) \\ &= I_{qn} + \underbrace{(I_{qn} + \tau(L_q^{-1} \otimes M^{-1}K))^{-1}}_{W_1^{-1}}\underbrace{(\widehat{U}_q \otimes I_n)}_{W_2},\end{aligned} \tag{14}$$

where $\widehat{U}_q$ is the strictly upper triangular part of U_q. Denote $Q = W_1^{-1}W_2$. Hence,

$$\|\mathcal{P}_L^{-1}\mathcal{A} - I_{qn}\| = \|Q\| \le \|W_1^{-1}\|\|W_2\|, \tag{15}$$

TABLE 1 Some characteristics of A_q^{-1}, L_q, and $\widehat{U}_q$ for different number of stages q, $S_q = \frac{1}{\ell} A_q^{-1} - I_q$, $\gamma = \left|\text{eig}_{\min}(S_q)\right|$.

	Number of stages q										
	2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
$\|\widehat{U}_q\|$	0.33	0.41	0.48	0.53	0.57	0.60	0.62	0.64	0.66	0.67	0.69
$\lambda_{\min}(L_q)$	1.5	2.07	2.94	3.58	4.41	5.08	5.88	6.65	7.35	8.04	8.82
$\lambda_{\max}(L_q)(=\ell)$	4	9	16	25	36	49	64	81	100	121	144
$\|S_q\|$	1.32	1.31	1.29	1.28	1.27	1.27	1.27	1.26	1.26	1.26	1.26
γ	0.61	0.60	0.71	0.74	0.79	0.82	0.84	0.86	0.87	0.88	0.89

which is predicted to be small since $\|\widehat{U}_q\|$ is reasonably less than 1. For illustration, in Table 1 we show the value of $\|\widehat{U}_q\|$ for a number of stages $2 \le q \le 12$. Further, if, for instance, K has a positive definite symmetric part, then $\|W_1^{-1}\|$ is likely smaller than unity and particularly small for large time-steps τ.

We next attempt to derive an upper bound of the spectral radius of the preconditioned matrix $\mathcal{P}_L^{-1}\mathcal{A}$. To this end, as an analysis tool we use the field of values $\mathcal{W}(A)$, defined as $\mathcal{W}(A) = \{\boldsymbol{x}^* A\boldsymbol{x} : \boldsymbol{x} \in \mathbb{C}, \|\boldsymbol{x}\| = 1\}$, where $\boldsymbol{x}^*$ denotes the conjugate transpose of the vector $\boldsymbol{x}$. Let A and B denote generic square matrices, $\mathcal{S}(A)$ be the spectrum of A and $\mathcal{W}(A)$ - its field of values, $\rho(A)$ and $\omega(A)$ be the spectral and the numerical radius of A, respectively, defined as $\rho(A) = \max\{|\lambda| : \lambda \in \text{spectrum of } A\}$ and $\omega(A) = \max\{|\boldsymbol{x}^* A\boldsymbol{x}| : \|\boldsymbol{x}\| = 1\}$. Let also $\sigma_1(A) \ge \sigma_2(A) \ge \ldots \ge \sigma_n(A)$ be the singular values of A. The following relations are known to hold:

(P1) $\rho(A) \le \omega(A) \le \sigma_1(A)$, compare, for example, Reference 60, p. 137.
(P2) $\frac{1}{2}\|A\| \le \omega(A) \le \|A\|$, compare, for example, Reference 60, p. 137.
(P3) $\omega(AB) \le 4\,\omega(A)\,\omega(B)$, compare Reference 61, Theorem 2.5-2.
(P4) For general matrices A and B there holds (cf. Reference 61 Theorem 2.4-1)

$$\rho(A^{-1}B) \subset \frac{\overline{\mathcal{W}(B)}}{\overline{\mathcal{W}(A)}} = \left\{ \frac{r_2}{r_1} : r_1 \in \overline{\mathcal{W}(A)},\ r_2 \in \overline{\mathcal{W}(B)} \right\},$$

where $\overline{\mathcal{W}(\cdot)}$ denotes the closure of $\mathcal{W}(\cdot)$.

(P5) Let $A \in \mathbb{C}^{m\times m}$ and $B \in \mathbb{C}^{n\times n}$ with eigenvalues $\{\lambda_i, i = 1, \ldots, m\}$ and $\{\mu_j, j = 1, \ldots, n\}$, correspondingly. Then, the spectrum of $A \otimes B$ consists of $\{\lambda_i\mu_j\}, i = 1, \ldots, m,\ j = 1, \ldots, n$. The same holds for the singular values, compare Reference 62 Theorem 4.2.12.

(P6) Denote the eigenvalues of $M^{-1}K$ by $0 < \theta_1 \le \ldots \le \theta_n$, where M and K are the matrices in (1).
The spectrum $\{\theta_j\}, j = 1, \ldots, n$ depends on the time-independent part of the equation at hand, whether it is of diffusion with or without anisotropy, convection-diffusion, or only of advection type, as well as of the chosen discretization in space. The mass matrix M is symmetric positive definite and in the case of a space operator of the form $-\Delta u + \boldsymbol{b}\cdot\nabla u$, K is also positive definite. For piece-wise linear finite element discretization in space with a mesh-size h and for arbitrary real vector $\boldsymbol{x}$, such that $\|\boldsymbol{x}\| = 1$, we can estimate θ_j from the following relations,

$$c_1 h^2 \le \boldsymbol{x}^T M\boldsymbol{x} \le c_2 h^2, \qquad d_1 h^2 \le \boldsymbol{x}^T K\boldsymbol{x} \le d_2, \qquad \widetilde{c}_1 \le \theta_j \le \widetilde{c}_2 h^{-2}. \tag{16}$$

Here $c_i > 0, d_i > 0, i = 1, 2$ and $\widetilde{c}_1 = d_1/c_2, \widetilde{c}_2 = d_2/c_1$ are constants, not depending on h. The estimates (16) of the eigenvalues of the global FEM matrices are expressed in terms of the extremal nonzero eigenvalues of the element stiffness and mass matrices and the maximal degree of the nodes in the discretization mesh, compare, for example, Reference 63.

We state the following conjecture.

Conjecture 1. *Consider the matrix $\mathcal{A} = A_q^{-1} \otimes M + \tau I_q \otimes K$, where K and M are positive definite matrices, corresponding to a finite element discretization of a second order elliptic operator. Further, let A_q be the matrix,*

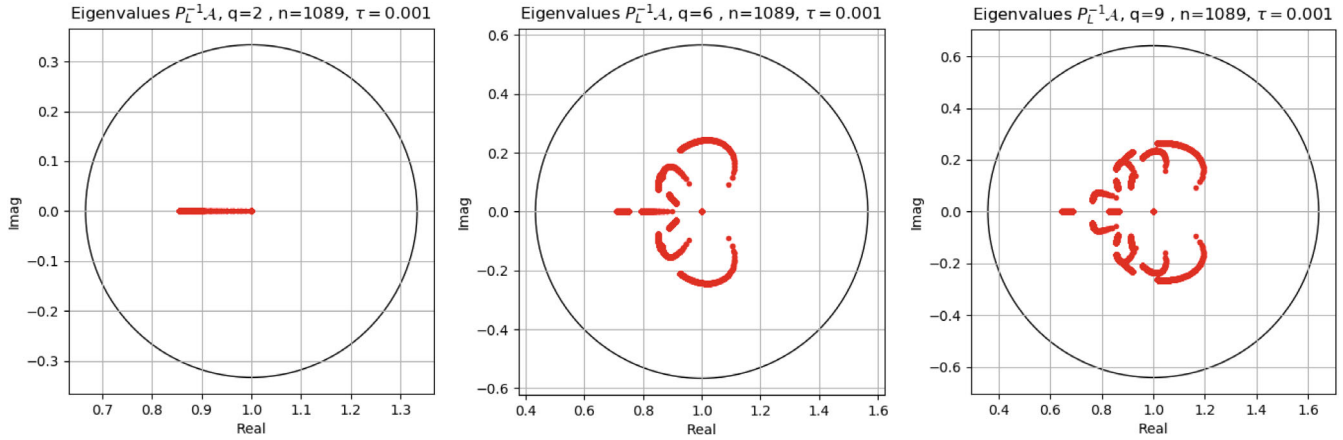


FIGURE 1 Problem 1: Eigenvalues of $\mathcal{P}_L^{-1}\mathcal{A}$ - $q = 2$, $q = 6$, $q = 9$, circle with radius $\|\hat{U}_q\|$ and centered at 1.

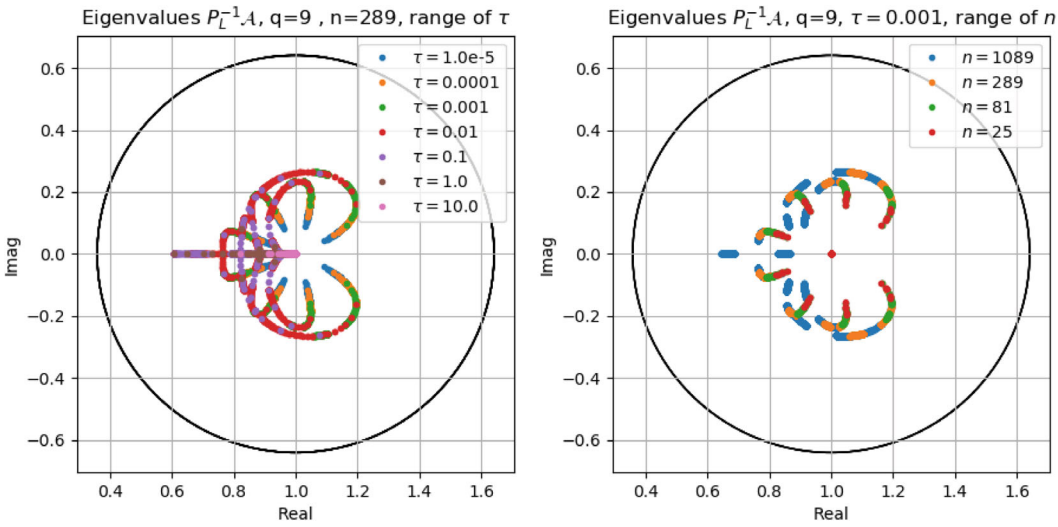


FIGURE 2 Problem 1: Eigenvalues of $\mathcal{P}_L^{-1}\mathcal{A}$ - $q = 9$ for a range of τ (left) and block dimension n (right), circle with radius $\|\hat{U}_9\|$ and centered at 1.

corresponding to a fully implicit Runge–Kutta method of Radau type with q stages and I_q be the identity matrix of order q. Let $A_q^{-1} = L_q U_q$ be the exact LU factorization of A_q^{-1}, such that U_q has unit diagonal. Denote $\widehat{U} = U_q - I_q$, strictly upper triangular and assume that $\|\widehat{U_q}\| < 1$. Define the matrix $\mathcal{P}_L = L_q \otimes M + \tau I_q \otimes K$. Consider the generalized eigenvalue problem

$$\mathcal{A}\boldsymbol{x} = \nu \mathcal{P}_L \boldsymbol{x}.$$

Then, all eigenvalues ν are contained in a circle, centered at 1 and with radius $\|\widehat{U}\|$, strictly less than one. Furthermore, there are at least n eigenvalues that are exactly equal to 1.

Remark 3. Conjecture 1 is supported by strong numerical evidence. Figures 1 and 2 illustrate the spectrum of $\mathcal{P}_L^{-1}\mathcal{A}$, showing that it is favorably clustered around 1. Figure 3 illustrates the spectrum for the pure convection case from Problem 3. Note that in the latter case the assumptions of the conjecture are not satisfied. The circular bound of the spectra is also shown in the figures. However, using various relations between spectral and numerical radius, norms, singular values, induced norm techniques, rigorous derivation has not been possible to show. We sketch below one of the possible approaches.

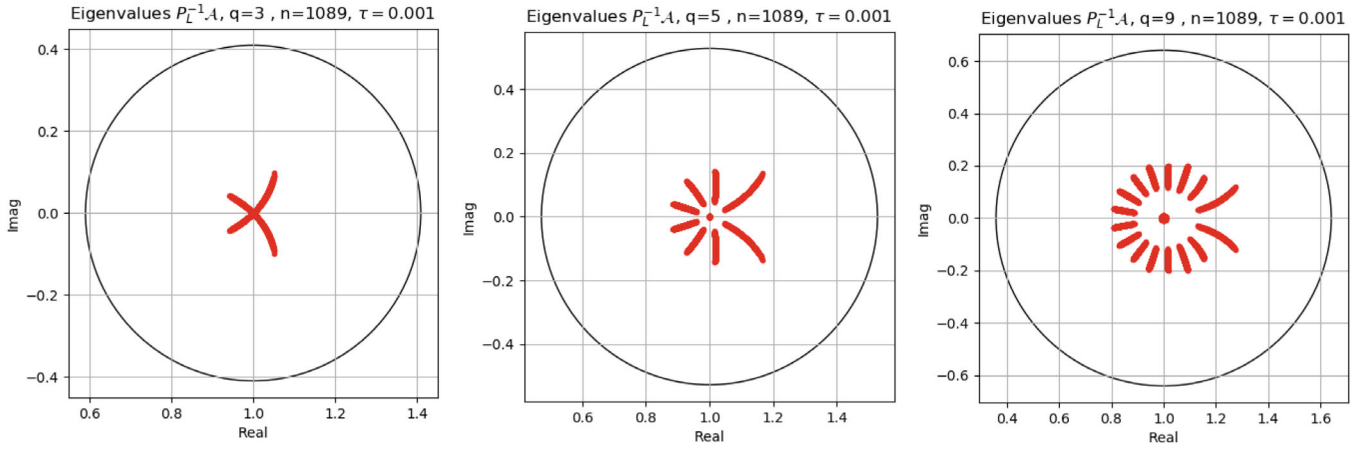


FIGURE 3 Problem 3: Eigenvalues of $\mathcal{P}_L^{-1}\mathcal{A}$ - $q = 3$, $q = 5$, $q = 9$, circle with radius $\|\hat{U}_q\|$ and centered at 1.

Consider the generalized eigenvalue problem $\mathcal{A}\boldsymbol{v} = \nu\mathcal{P}_L\boldsymbol{v}$. and transform it as

$$(\mathcal{A} - \mathcal{P}_L)\boldsymbol{v} = \mu\mathcal{P}_L\boldsymbol{v} \tag{17}$$

with $\mu = \nu - 1$. As $\mathcal{A} - \mathcal{P}_L = L_q\hat{U}_q \otimes M$, then μ is an eigenvalue of $Q_L\boldsymbol{v} = \mu\boldsymbol{v}$, where

$$Q_L \equiv (L_q \otimes M + \tau I_q \otimes K)^{-1}(L_q\hat{U}_q \otimes M) = \underbrace{(I_{qn} + \tau(L_q^{-1} \otimes M^{-1}K))^{-1}}_{W_1^{-1}}\underbrace{(\hat{U}_q \otimes I_n)}_{W_2}, \tag{18}$$

that is, $Q_L = W_1^{-1}W_2$. A straightforward computation shows that the matrix Q_L has one zero block-column of size $qn \times n$. Thus, n of the eigenvalues ν are equal to one. To estimate the rest $(q-1)n$ eigenvalues we want to bound $\rho(Q_L)$, or $\omega(Q_L)$.

As the matrices W_1^{-1} and $\hat{U}$ are nonsymmetric and do not commute, we estimate $\omega(Q_L)$ applying (P3) and we obtain that $\omega(Q_L) \leq 4\ \omega(W_1^{-1})\|\hat{U}_q\|$. We note that $\|\hat{U}_q\| < 1$. Even more, from Table 1 one can deduce that for any q $\|\hat{U}_q\| < \frac{3}{4}$.

Then, even though, based on estimating the eigenvalues of $M^{-1}K$ one can show that $\omega\left(I_{qn} + \tau L_q^{-1} \otimes M^{-1}K\right)^{-1} < 1$ it does not suffice to ensure that $\rho(Q_L) \leq 3\omega\left(W_1^{-1}\right)\|\hat{U}_q\| \leq 1$.

We include a proof that Conjecture 1 holds true for $q = 2$.

Theorem 1. *Let K and M be symmetric positive definite matrices and $\tau > 0$. For $q = 2$, the eigenvalues of $\mathcal{P}_L^{-1}\mathcal{A}$ lie in a circle in the complex plane centered at 1 with radius $\|\hat{U}_2\|$.*

Proof. We apply (P4) to bound $\rho(W_1^{-1}W_2)$ and, thus, we need to find the smallest $|r_1|$ in $\overline{\mathcal{W}(W_1)}$ and the largest $|r_2|$ in $\overline{\mathcal{W}(W_2)}$ It is straightforwardly seen that

$$|r_2| = \omega(W_2) \leq \|W_2\| = \|\hat{U}_q\|.$$

To estimate the smallest $|r_1|$ we use the relations

$$\mathcal{W}(W_1) = \mathcal{W}\left(\left(I_{qn} + \tau\left(L_q^{-1} \otimes M^{-1}K\right)\right)\right) \subset \mathcal{W}(I_{qn}) + \mathcal{W}\left(\tau\left(L_q^{-1} \otimes M^{-1}K\right)\right),$$

where the addition denotes a sumset. Next, since $\mathcal{W}(I_{qn}) = \{1\}$ it follows that

$$\mathcal{W}(W_1) \subset \mathcal{W}(I_{qn}) + \mathcal{W}\left(\tau\left(L_q^{-1} \otimes M^{-1}K\right)\right) = \left\{1 + r\ ; r \in \mathcal{W}\left(\tau\left(L_q^{-1} \otimes M^{-1}K\right)\right)\right\}.$$

Given that $\tau > 0$, to estimate the smallest $|r_1|$ it is enough to show that all $r \in \mathcal{W}((L_q^{-1} \otimes M^{-1}K))\}$ are nonnegative, that is, it suffices to show

$$\inf_{\|\boldsymbol{x}\|=1} \boldsymbol{x}^*(L_q^{-1} \otimes M^{-1}K)\boldsymbol{x} \geq 0. \tag{19}$$

Since M is symmetric and positive definite, the following similarity relation holds true,

$$\left(L_q^{-1} \otimes M^{-1}K\right) = \left(I_q \otimes M^{-\frac{1}{2}}\right)\left(L_q^{-1} \otimes M^{-\frac{1}{2}}KM^{-\frac{1}{2}}\right)\left(I_q \otimes M^{\frac{1}{2}}\right).$$

Therefore, (19) is equivalent to $\inf_{\|\boldsymbol{x}\|=1} \boldsymbol{x}^*\left(L_q^{-1} \otimes M^{-\frac{1}{2}}KM^{-\frac{1}{2}}\right)\boldsymbol{x} \geq 0$, where $\widetilde{K} \equiv M^{-\frac{1}{2}}KM^{-\frac{1}{2}}$ is symmetric positive definite. Denote $L_q^{-1} = \{l_{ij}\}$, $i,j = 1, \ldots, q$ and recall, that $l_{ij} > 0$, compare Remark 1. Let $\boldsymbol{x}^* = [\boldsymbol{x}_1^*, \ldots, \boldsymbol{x}_q^*], \boldsymbol{x}_i \in \mathbb{C}^{n,1}$ and $S \equiv \boldsymbol{x}^*\left(L_q^{-1} \otimes \widetilde{K}\right)\boldsymbol{x}$. Then

$$S = \sum_{k=1}^{q}\left(\sum_{j=1}^{k} l_{k,j}\boldsymbol{x}_k^*\widetilde{K}\boldsymbol{x}_j\right) = \underbrace{\sum_{k=2}^{q}\left(\sum_{j=1,j\neq k}^{k} l_{k,j}\langle \boldsymbol{x}_k, \boldsymbol{x}_j\rangle_{\widetilde{K}}\right)}_{S_{od}} + \underbrace{\sum_{k=1}^{q} l_{k,k}\|\boldsymbol{x}_k\|_{\widetilde{K}}^2}_{S_d>0}. \tag{20}$$

As $\widetilde{K}$ is symmetric positive definite it induces a norm and an inner product. Therefore, in order to show that $S \geq 0$, it is enough to have that $|S_{od}| \leq S_d$. We have:

$$|S_{od}| \leq \sum_{k=2}^{q}\left(\sum_{j=1,j\neq k}^{k} l_{k,j}|\langle \boldsymbol{x}_k, \boldsymbol{x}_j\rangle_{\widetilde{K}}|\right) \leq \sum_{k=2}^{q}\left(\sum_{j=1,j\neq k}^{k} l_{k,j}\|\boldsymbol{x}_k\|_{\widetilde{K}}\|\boldsymbol{x}_j\|_{\widetilde{K}}\right),$$

where we use the Cauchy–Schwarz inequality on the right relation. Now it suffices to show

$$\sum_{k=2}^{q}\left(\sum_{j=1,j\neq k}^{k} l_{k,j}\|\boldsymbol{x}_k\|_{\widetilde{K}}\|\boldsymbol{x}_j\|_{\widetilde{K}}\right) \leq \sum_{k=1}^{q} l_{k,k}\|\boldsymbol{x}_k\|_{\widetilde{K}}^2. \tag{21}$$

For $q = 2$, in order for (21) to hold, we need to show that $l_{21}\|\boldsymbol{x}_2\|\|\boldsymbol{x}_1\| \leq l_{11}\|\boldsymbol{x}_1\|^2 + l_{22}\|\boldsymbol{x}_2\|^2$. If either $\|\boldsymbol{x}_i\| = 0$ this clearly holds. Assuming $\|\boldsymbol{x}_i\| \neq 0$, we denote $z = \|\boldsymbol{x}_1\|/\|\boldsymbol{x}_2\| > 0$ and dividing both sides by $l_{21}\|\boldsymbol{x}_2\|\|\boldsymbol{x}_1\|$, the relation to be shown reads

$$1 \leq \frac{l_{11}\|\boldsymbol{x}_1\|^2}{l_{21}\|\boldsymbol{x}_2\|\|\boldsymbol{x}_1\|} + \frac{l_{22}\|\boldsymbol{x}_2\|^2}{l_{21}\|\boldsymbol{x}_2\|\|\boldsymbol{x}_1\|} = \frac{l_{11}\|\boldsymbol{x}_1\|}{l_{21}\|\boldsymbol{x}_2\|} + \frac{l_{22}\|\boldsymbol{x}_2\|}{l_{21}\|\boldsymbol{x}_1\|} = \frac{l_{11}}{l_{21}}z + \frac{l_{22}}{l_{21}}\frac{1}{z}.$$

The latter is equivalent to

$$p(z) \equiv \frac{l_{11}}{l_{21}}z^2 + \frac{l_{22}}{l_{21}} - z \geq 0.$$

Inserting $l_{11} = 2/3,\ l_{21} = 3/4,\ l_{22} = 1/4$ we obtain that the inequality

$$p(z) = \frac{8}{9}z^2 + \frac{1}{3} - z \geq 0$$

holds true as $p(z)$ has a global minimum at $p(9/16) = 5/96 > 0$. ∎

5.2 | Spectral properties of the preconditioned matrix $\mathcal{P}_D^{-1}\mathcal{A}$

For the spectrum of the preconditioned matrix $\mathcal{P}_D^{-1}\mathcal{A}$ the following result holds true.

Theorem 2. *Consider the matrix $\mathcal{A} = A_q^{-1} \otimes M + \tau I_q \otimes K$, where K and M are positive definite matrices, corresponding to a finite element discretization of a second order elliptic operator. Further, let A_q be the matrix, corresponding to a fully implicit Runge–Kutta method of Radau type with q stages and I_q be the identity matrix of order q. Let $A_q^{-1} = L_q U_q$ be the exact LU factorization of A_q^{-1}, such that U_q has unit diagonal. Let ℓ be the largest eigenvalue of L_q. Define the matrix $\mathcal{P}_D = \ell I_q \otimes M + \tau I_q \otimes K$ and consider the generalized eigenvalue problem*

$$\mathcal{A}\boldsymbol{x} = \nu \mathcal{P}_D \boldsymbol{x}.$$

Then, all eigenvalues ν are contained in a circle, centered at 1 and with radius $\delta_q > 1$, where δ_q depends only on q. In addition, the real parts of the eigenvalues are larger than a constant $C_q > 0$, depending only of the number of stages q.

Proof. To analyze the spectrum of the preconditioned matrix $\mathcal{P}_D^{-1}\mathcal{A}$ we consider the generalized eigenvalue problem $\mathcal{A}\boldsymbol{v} = \nu \mathcal{P}_D \boldsymbol{v}$, transform it to $(\mathcal{A} - \mathcal{P}_D)\boldsymbol{v} = (\nu - 1)\mathcal{P}_D \boldsymbol{v}$ and analyze $\mu = \nu - 1$, which is an eigenvalue of $((A_q^{-1} - \ell I_q) \otimes M)\boldsymbol{v} = \mu(\ell I_q \otimes M + \tau I_q \otimes K)\boldsymbol{v}$. In order to derive spectral bounds we rewrite the matrices in matrix-tensor product form:

$$\begin{aligned} Q_D &\equiv (\ell I_q \otimes M + \tau I_q \otimes K)^{-1} \left(\left(A_q^{-1} - \ell I_q \right) \otimes M \right) \\ &= \left(I_q \otimes (\ell M + \tau K)^{-1} \right) \left(\left(A_q^{-1} - \ell I_q \right) \otimes M \right) \\ &= \left(\tfrac{1}{\ell} A_q^{-1} - I_q \right) \otimes \left(I_n + \tfrac{\tau}{\ell} M^{-1} K \right)^{-1} \equiv Z_1 \otimes Z_2^{-1}. \end{aligned} \tag{22}$$

Then, using (P5) we obtain

$$\omega(Q_D) \le \omega(Z_1)\omega\left(Z_2^{-1}\right) \le \| \tfrac{1}{\ell} A_q^{-1} - I_q \| \omega\left(\left(I_{qn} + \tfrac{\tau}{\ell} M^{-1} K \right)^{-1} \right) < \delta_q. \tag{23}$$

From Table 1 we see that $1 < \delta_q < 1.5$ for $q \le 12$. Since the bound in (23) is larger than 1 and we want to estimate $\nu = \mu + 1$, we derive a bound of the real part of ν.

Due to the matrix-tensor form of $Q_D = Z_1 \otimes Z_2^{-1}$, using (P5), we have that $\mu = \mu_1 \mu_2$, where μ_1 is an eigenvalue of Z_1 and μ_2 is an eigenvalue of Z_2^{-1}. From (P6) we are in a position to estimate μ_2, which are real and positive, namely,

$$\frac{\ell}{\ell + \widetilde{c}_2 \tau\, h^{-2}} \le \mu_2 \le \frac{\ell}{\ell + \widetilde{c}_2 \tau} < 1.$$

Further, from Table 1 we see that, at least up to $q = 12$, $|\mu_1| < g_q < 1$. Therefore, $Re(\mu_1) < g_q$ and $Re(\nu) = Re(1 + \mu_1\mu_2) > 1 - Re(\mu_1) > 1 - g_q \equiv C_q > 0$. ∎

The spectrum of $\mathcal{P}_D^{-1}\mathcal{A}$ is illustrated in Figure 4.

6 | IMPLEMENTATION ASPECTS

6.1 | Choice of inner solvers

As already mentioned, the systems (4) and (5) are in general very large scale, of order qn but with the approach taken here we solve systems only of size n. In typical problems, in particular for three dimensional problems in space, n itself is very large. We assume that we can either construct a feasible approximation of the diagonal blocks $\lambda_j M + \tau K$, $j = 1, 2, \ldots, q$ or can solve the inner systems with some efficient iterative solution method. A suitable approximation could for instance be based on a modified incomplete factorization, see References 64 and 65 with sparse matrix factors. When solving systems with the diagonal blocks we can apply some algebraic multilevel technique, see, for example, References 66, 67 and 60, or an algebraic multigrid (AMG) solver, which we utilize in this study, using library-provided AMG implementations.

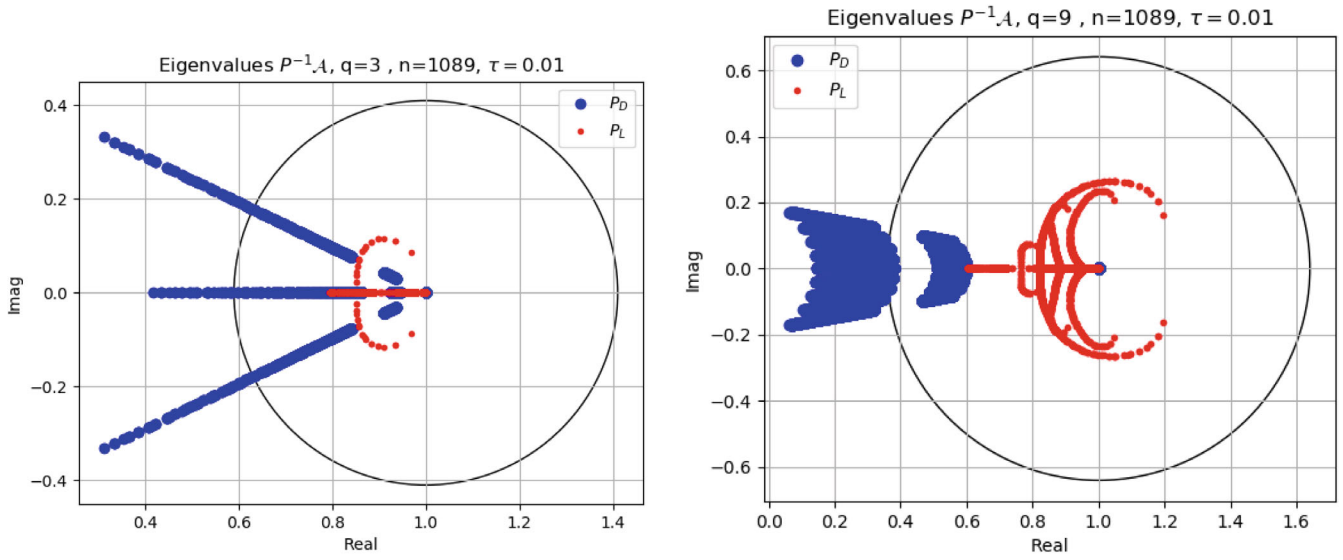


FIGURE 4 Problem 1: Eigenvalues of $\mathcal{P}_D^{-1}\mathcal{A}$ (blue) and of $\mathcal{P}_L^{-1}\mathcal{A}$ (red)—$q = 3, 9$, circle with radius $\|\hat{U}_q\|$ and centered at 1.

6.2 | Parallelization aspects

As the target solution method is a preconditioned Krylov Subspace iteration, the two major ingredients are the matrix-vector multiplication with the matrix $\mathcal{A}$ and the solution of linear systems with the preconditioning matrix $\mathcal{P}_L$ or $\mathcal{P}_D$. The matrices $\mathcal{A}$, $\mathcal{P}_L$, and $\mathcal{P}_D$ are of dimension $qn \times qn$. They are never explicitly stored but rather we utilize their structure to implement the above two operations efficiently in parallel.

We aim at distributed memory MPI-based implementation of the computations. In the chosen parallel computing environment there are two general strategies to administer the fully implicit IRK method and the preconditioner $\mathcal{P}_L$ in (12).

The first strategy is to distribute the space discretization mesh among a number of processes. This entails that the solution of each system on the diagonal of $\mathcal{P}_d$ is done in parallel in space, using well-known parallelizable methods, such as multilevel, multigrid, domain-decomposition techniques and so forth implemented and optimized in some of the established scientific computing libraries. This is the implementation chosen for the numerical tests in this work and we do not implement parallelism across the stages.

When it comes to implementing simultaneous parallelization across stages and in space, different strategies could be employed. This aspect of the implementation is a topic of an independent study.[26] The strategy, chosen there, is based on allocating q groups of processes and within group carrying out space parallelism, using matrix-free operations and matrix-vector multiplications in a block-fashion. The tests include discretization with higher order finite elements, comparisons between a stage-parallel and stage-serial implementation as well as between the performance of $\mathcal{P}_L$ using only real arithmetic with that of the complex factorization from Reference 39.

7 | NUMERICAL TESTS

The parallel tests are run on the Rackham cluster at the Uppsala Multidisciplinary Center for Advanced Computational Science (UPPMAX). Rackham consists of 486 nodes, each node having two 10-core Intel Xeon E5 2630 v4 at 2.20 GHz/core. We use nodes with 128 GB of memory. The test problems are implemented in the `deal.II` FEM library,[68] interfacing with PETSc[69] to utilize the available preconditioned iterative methods. The space discretization for Problems 1 and 2 is done by piece-wise linear and for Problem 3—by third-order conforming finite element basis functions on quadrilateral meshes.

The arising systems for the stage variables are solved by an inner-outer solution procedure. Due to the latter the outer solver must allow for variable preconditioners. Our choice of outer solution method is GCR

(Algorithm 3.1 in Reference 57), which, together with GCG and FGMRES, is capable of handling variable preconditioning. For Problems 1 and 2 the inner block systems in $\mathcal{P}_L$ and $\mathcal{P}_D$ are solved by AMG-preconditioned CG (P1)/GCR (P2). For Problem 3 the blocks are solved using the MUMPS sparse direct solver.

The tests, presented here, utilize only space parallelism, that is, the block systems in the preconditioners are solved one after another. The full stage- and space-parallel implementation is outlined in Section 6.2 and implemented and tested in Reference 26.

The performance of the IRK method and the proposed preconditioner are illustrated using the following three test problems.

Problem 1 (Heat equation in two-space dimensions). Consider the equation

$$\begin{aligned}\frac{\partial u(x,y,t)}{\partial t} - \Delta u(x,y,t) &= f(x,y,t) \text{ in } \Omega = (0,1)^2,\ t \in [0,T]\\ u(x,y,0) &= u_{ex}(x,y,0),\quad (x,y) \in \Omega\\ u(x,y,t) &= u_{ex}(x,y,t),\quad (x,y) \in \partial\Omega,\end{aligned}$$

and choose $f(x,y,t)$ to corresponds to the analytical solution

$$u_{ex}(x,y,t) = \sin(a_x\pi x)\sin(a_y\pi y)(1+\sin(\pi t))e^{-a_t t},$$

$$\begin{aligned}f(x,y,t) =& \sin(a_x\pi x)\sin(a_y\pi y)(\pi\cos(\pi t) - a_t(1+\sin(\pi t)))e^{-a_t t}\\ &+ (a_x^2 + a_y^2)\pi^2\sin(a_x\pi x)\sin(a_y\pi y)(1+\sin(\pi t))e^{-a_t t}.\end{aligned}$$

For the tests we choose $a_x = a_y = 2$ and $a_t = 0.5$. For the outer solver we use a variant of GCR with relative stopping criteria 10^{-8}. For the q block-systems in the preconditioner we solve using CG preconditioned by an AMG, with a relative stopping criteria of 10^{-6}. We fix $\tau = 0.1$ and take five time-steps. The tests are with two and nine stages.

Problem 2 (Similar to Reference 55). Consider the two-dimensional time-dependent convection-diffusion equation

$$\frac{\partial u(x,y,t)}{\partial t} + s(t)(-\varepsilon\Delta u + \boldsymbol{b}\cdot\nabla u - f) = 0 \text{ in } \Omega = (0,1)^2,\ t > 0, \tag{24}$$

with $\varepsilon = 1$, initial condition $u(x,y,0) = u_0$ – the pyramid function shown in Figure 5, boundary conditions

$$\begin{cases} u = 0 & \text{on } y = 0, y = 1\\ u = g(x,y,t) & \text{on } x = 1,\\ \frac{\partial u}{\partial n} = 0 & \text{on } x = 0,\end{cases}$$

$\boldsymbol{b}(x,y) = [-\ell, 0]$, $\ell > 1$, $\sigma(t) = 1 + \frac{2}{5}\sin(k\pi t)$, $g(x,y,t) = 2\ell y(1-y)\sin(k\pi t)$ and $f(x,y,t) = 2e^{-\ell x}$. As we deal with non-homogeneous and time-dependent boundary conditions, we follow the general practice to construct a partial solution $\psi(x,y,t)$ that satisfies the boundary conditions and then reformulate the problem to find $v = u - \psi$. In order to homogenize the boundary conditions we choose $\psi(x,y,t) = e^{-\ell x}x^2y(1-y)\sin(k\pi t)$. We find then that the function $v(x,y,t) = u(x,y,t) - \psi(x,y,t)$ is the solution of the initial boundary problem

$$\frac{\partial v(x,y,t)}{\partial t} + s(t)(-\varepsilon\Delta v + \boldsymbol{b}\cdot\nabla v - \widetilde{f}) = 0 \text{ in } \Omega = (0,1)^2,\ t > 0,$$

with $\widetilde{f} = f + \frac{1}{s(t)}\frac{\partial\psi}{\partial t} - \Delta\psi + b\cdot\nabla\psi$, boundary conditions $v = 0$ on $y = 0, y = 1, x = 1$ and $\frac{\partial v}{\partial n} = 0$ on $x = 0$, initial condition $v(x,y,0) = u_0(x,y) - \psi(x,y,0) = u_0(x,y)$. In detail,

$$\widetilde{f}(x,y,t) = -2e^{-\ell x} + \frac{k\pi e^{-\ell x}x^2y(1-y)\cos(k\pi t)}{1+\frac{2}{5}\sin(k\pi t)} - \left[(\ell^2x^2 - 4\ell x + 2)y(1-y) - 2x^2\right]e^{-\ell x}\sin(k\pi t).$$

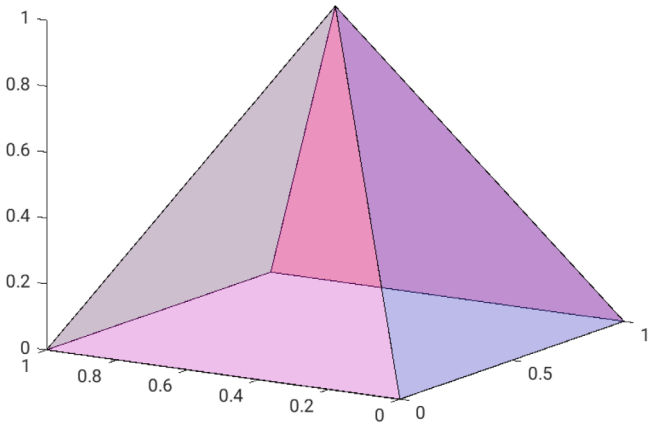


FIGURE 5 Problem 2: Initial condition.

The additional complication with the σ-scaling is taken into account when constructing the preconditioner as follows. The system equation, analogous to (4), now reads

$$(I_q \otimes M + \tau \Sigma_q A_q \otimes K)k = \hat{\mathbf{f}},$$

where Σ_q is a diagonal matrix, containing the values of σ in the integration points. Multiplying the latter equation by $A_q^{-1}\Sigma_q^{-1}$ from the left we obtain

$$(A_q^{-1}\Sigma_q^{-1} \otimes M + \tau I_q \otimes K)\mathbf{k} = (A_q^{-1}\Sigma_q^{-1} \otimes I_n)\hat{\mathbf{f}}.$$

As for Problem 1, we then select the preconditioner to be $\mathcal{P}_L := (L\Sigma_q^{-1} \otimes M + \tau I_q \otimes K)$. Finally we decompose $L_q\Sigma_q^{-1} = T\Lambda T^{-1}$ and obtain an analogous form of the preconditioner used in Problem 1 with the difference that the eigenvalue decomposition is computed in every time step. The cost of this is low, however, as the matrix in question is of size $q \times q$. The outer solver is GCR with stopping criteria $\|Ax - b\|_2 < 10^{-8}$. For the q block-systems in the preconditioner we now use GCR preconditioned by an AMG with stopping criteria $\|Ax - b\|_2 < 10^{-6}$.

Problem 3 (70). Consider the convection-diffusion equation (24) with $s(t) = 1$, $\Omega = [-1, 1]^2$, $f = 0$ and $\boldsymbol{b} = [-4y, 4x]^T$ and initial condition—the Gaussian pulse

$$u(x, y, 0) = e^{-\frac{(x+x_c)^2+(y-y_c)^2}{2\sigma^2}}.$$

Here x_c, y_c, and σ are the center of the pulse and the standard deviations, respectively. The corresponding analytical solution for a constant diffusion ε is given by

$$u_a(x, y, t) = \frac{2\sigma^2}{2\sigma^2 + 4\varepsilon t}\, e^{-\frac{(\bar{x}+x_c)^2+(\bar{y}-y_c)^2}{2\sigma^2+4\varepsilon t}},$$

where $\bar{x} = x\cos(4t) + y\sin(4t)$ and $\bar{y} = -x\sin(4t) + y\cos(4t)$. The time interval is $[0, \frac{\pi}{2}]$, equal to the time for one complete rotation of the pulse.

As noted in Reference 70, the behavior of this problem changes from advection-dominated in most of the domain to diffusion-dominated close to the origin. We solve this problem in the purely advection case, namely, for the numerical experiments we choose $\varepsilon = 0, x_c = -0.5, y_c = 0$, and $\sigma = 0.0447$. As the value of the analytical solution on the boundary of the domain is less than the machine accuracy we choose homogeneous boundary conditions. The inner multigrid solver used in Problems 1 and 2 is not suitable for the pure advection case and to keep this work contained and focused on the proposed block preconditioners, we replace the inner multigrid-based solver by a parallel direct solver and present the outer iterations.

TABLE 2 Problem 1 $\mathcal{P}_L$: Parallel run times and errors for $q = 2$, solving for 5 time steps with $\tau = 0.1$.

No. CPUs	Total runtime	Av. out. iter.	Av. in. iter.	$\|\|u_c(\tau_F) - u_a(\tau_F)\|\|_\infty$	$\|\|u_c(\tau_F) - u_a(\tau_F)\|\|_2/n_{dofs}$
Block dimension 1,050, 625, full system dimension 2,101, 250					
20	10	3	7	7.014×10^{-5}	3.418×10^{-8}
Block dimension 4,198, 401, full system dimension 8,396, 802					
20	38	3	7	7.382×10^{-5}	1.801×10^{-8}
Block-dimension 16,785, 409, full system dimension 33,570, 818					
100	39	3	8	7.474×10^{-5}	9.119×10^{-9}
Block dimension 67,125, 249, full system dimension 134,250, 498					
100	203	3	8	7.497×10^{-5}	4.575×10^{-9}
200	130	3	8	7.497×10^{-5}	4.575×10^{-9}
300	99	3	8	7.497×10^{-5}	4.575×10^{-9}
Block dimension 268,468, 225, full system dimension 536,936, 450					
600	143	3	9	7.503×10^{-5}	2.289×10^{-9}

Note: Errors evaluated at $\tau_F = 0.5$.

TABLE 3 Problem 1 $\mathcal{P}_L$: Parallel run times and errors for $q = 9$, solving for 5 time steps with $\tau = 0.1$.

No. CPUs	Total runtime	Av. out. iter.	Av. in. iter.	$\|\|u_c(\tau_F) - u_a(\tau_F)\|\|_\infty$	$\|\|u_c(\tau_F) - u_a(\tau_F)\|\|_2/n_{dofs}$
Block dimension 1,050, 625, full system dimension 9,455, 625					
20	119	10	6	4.932×10^{-6}	2.403×10^{-9}
Block dimension 4,198, 401, full system dimension 37,785, 609					
20	496	9	6	1.246×10^{-6}	3.039×10^{-10}
Block-dimension 16,785, 409, full system dimension 151,068, 681					
100	464	10	7	3.246×10^{-7}	3.960×10^{-11}
Block dimension 67,125, 249, full system dimension 604,127, 241					
100	2246	10	7	9.435×10^{-8}	5.752×10^{-12}
200	1414	10	7	9.437×10^{-8}	5.749×10^{-12}
300	762	10	8	9.441×10^{-8}	5.753×10^{-12}
Block dimension 268,468, 225, full system dimension 2,416, 214,025					
600	1994	10	8	3.707×10^{-8}	1.115×10^{-12}

Note: Errors evaluated at $\tau_F = 0.5$.

7.1 | Results for Problems 1 and 2

For Problem 1 we carry out tests for the parallel performance of the preconditioners $\mathcal{P}_L$ and $\mathcal{P}_D$. The results for $\mathcal{P}_L$ are found in Table 2 for $q = 2$ and in Table 3—for $q = 9$. The corresponding results for $\mathcal{P}_D$ are shown in Tables 4 and 5.

From the timing results for $q = 2$ in Table 2, column 2, we see that speedup in the strong scaling, also referred to as the the fixed-size speedup, is found to be $203/130 \approx 1.6$, compared to the ideal factor 2 and $99/203 \approx 2$ compared to ideal factor 3. Analogously, from the timing results for $q = 9$ in Table 3, column 2, we compute the fixed-size speedup as $2246/1414 \approx 1.6$, compared to the ideal factor 2 and $2246/762 \approx 2.9$ close to the ideal factor 3. Comparing the performance of $\mathcal{P}_L$ from Tables 2 and 3 with that of $\mathcal{P}_D$ in Tables 4 and 5, we see that $\mathcal{P}_D$ does not require more outer iterations and is superior to $\mathcal{P}_L$ in terms of total solution time. The main reason for this is that, as $\mathcal{P}_D$ assumes a constant block-diagonal, it needs only to set up a single AMG in comparison to the q different AMG solvers for $\mathcal{P}_L$. We find that using constant diagonal blocks does not harm the convergence of the outer iterative solver in the case of Problem 1. The effect of saving

TABLE 4 Problem 1 $\mathcal{P}_D$: Parallel run times and errors for $q = 2$, solving for 5 time steps with $\tau = 0.1$.

No. CPUs	Total runtime	Av. out. iter.	Av. in. iter.	$\|u_c(\tau_F) - u_a(\tau_F)\|_\infty$	$\|u_c(\tau_F) - u_a(\tau_F)\|_2/N$
Block dimension 1,050, 625, full system dimension 9,455, 625					
20	5	3	7	7.013×10^{-5}	3.418×10^{-8}
Block dimension 4,198, 401, full system dimension 37,785, 609					
20	20	3	7	7.382×10^{-5}	1.801×10^{-8}
Block-dimension 16,785, 409, full system dimension 151,068, 681					
100	23	3	8	7.474×10^{-5}	9.119×10^{-9}
Block dimension 67,125, 249, full system dimension 604,127, 241					
100	80	3	8	7.497×10^{-5}	4.575×10^{-9}
200	46	3	8	7.497×10^{-5}	4.575×10^{-9}
300	38	3	8	7.497×10^{-5}	4.575×10^{-9}
Block dimension 268,468, 225, full system dimension 2,416, 214,025					
600	73	3	9	7.503×10^{-5}	2.289×10^{-9}

Note: Errors evaluated at $\tau_F = 0.5$.

TABLE 5 Problem 1 $\mathcal{P}_D$: Parallel run times and errors for $q = 9$, solving for 5 time steps with $\tau = 0.1$.

No. CPUs	Total runtime	Av. out. iter.	Av. in. iter.	$\|u_c(\tau_F) - u_a(\tau_F)\|_\infty$	$\|u_c(\tau_F) - u_a(\tau_F)\|_2/N$
Block dimension 1,050, 625, full system dimension 9,455, 625					
20	53	11	6	4.931×10^{-6}	2.403×10^{-9}
Block dimension 4,198, 401, full system dimension 37,785, 609					
20	223	11	6	1.246×10^{-6}	3.039×10^{-10}
Block-dimension 16,785, 409, full system dimension 151,068, 681					
100	215	10	7	3.246×10^{-7}	3.960×10^{-11}
Block dimension 67,125, 249, full system dimension 604,127, 241					
100	867	10	7	9.434×10^{-8}	5.751×10^{-12}
200	453	10	7	9.437×10^{-8}	5.749×10^{-12}
300	372	10	8	9.441×10^{-8}	5.753×10^{-12}
Block dimension 268,468, 225, full system dimension 2,416, 214,025					
600	737	10	8	3.707×10^{-8}	1.115×10^{-12}

Note: Errors evaluated at $\tau_F = 0.5$.

is less pronounced in the case of full stage parallelism as the implementation needs q separate multigrid solvers by construction. As shown in Reference 26 the benefits of stage-parallelism are seen at the scaling limit, hence in many settings a stage-serial implementation may be preferable and $\mathcal{P}_D$ may be preferred over $\mathcal{P}_L$ due to its lower computational cost.

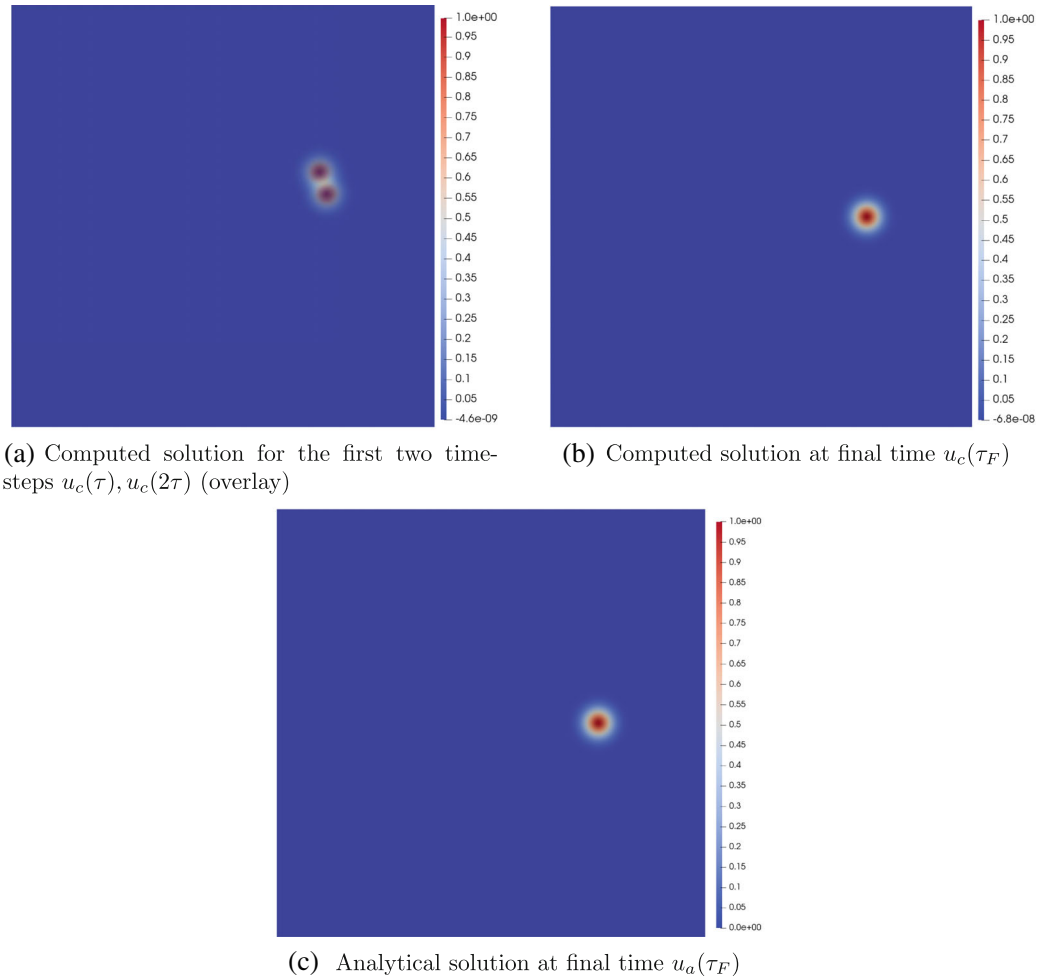
Problem 2 is tested only for $\mathcal{P}_L$. The corresponding results for $q = 2$ are shown in Table 6.

7.2 | Results for Problem 3

As we consider the pure advection case with $\varepsilon = 0$, CG/AMG is no longer an efficient inner solver. For the purpose of testing the outer preconditioner, as an inner solver we use here a sparse direct solver. As in Reference 41 we use Q_3 elements in space, select a fixed time-step and a number of stages, and then choose h such that $\tau^{2q-1} \approx h^4$. Plots of the solution of this problem are shown in Figure 6. The outer iterations are given in Table 7.

TABLE 6 Problem 2: Parallel run times and iteration counts using $\mathcal{P}_L$.

No. CPUs	Total runtime	Av. out. iter.	Av. in. iter.	No. CPUs	Total runtime	Av. out. iter.	Av. in. iter.
$q = 2, l = 2, k = 5, \tau = 0.1$				$q = 4, l = 10, k = 10, \tau = 0.001$			
Block dim. 4 198 401				Block dim. 4 198 401			
20	61	5	11	20	201	9	20
Block dim. 16 785 409				Block dim. 16 785 409			
100	59	4	12	100	178	9	20
Block dim. 67 125 249				Block dim. 67 125 249			
100	217	4	13	100	724	8	21
200	124	4	13	200	357	9	21
300	98	4	12	300	299	9	21
Block dim. 268 468 225				Block dim. 268 468 225			
600	187	4	14	600	609	8	22



(a) Computed solution for the first two time-steps $u_c(\tau), u_c(2\tau)$ (overlay)

(b) Computed solution at final time $u_c(\tau_F)$

(c) Analytical solution at final time $u_a(\tau_F)$

FIGURE 6 Problem 3: Plots of the exact and the numerically computed solutions with $q = 10$, $\tau = \pi/60$, $\tau_F = \pi/2$. Q_1 space discretization using with 66,049 DoFs, 35 avg. outer iterations.

TABLE 7 Problem 3: Outer iterations for the pure advection case using a direct solver as inner solver, space discretized by Q_3 elements.

Stages	τ	iter.	Precon.	Block dimension	# cells
$q = 3$	$\pi/60$	19	$\mathcal{P}_L$	37,249	4096
$q = 4$	$\pi/60$	27	$\mathcal{P}_L$	591,361	65,536
$q = 5$	$\pi/60$	33	$\mathcal{P}_L$	37,761, 025	4,194,304
$q = 3$	$\pi/60$	47	$\mathcal{P}_D$	37,249	4096
$q = 4$	$\pi/60$	58	$\mathcal{P}_D$	591,361	65,536
$q = 5$	$\pi/60$	>60	$\mathcal{P}_D$	37,761, 025	4,194,304

In contrast to the results for Problem 1, Table 7 shows that $\mathcal{P}_D$ exhibits a significantly worse convergence for the more difficult advection case. The iteration counts for both preconditioners grow with the number of stages but $\mathcal{P}_L$ reaches convergence in substantially fewer iterations.

Remark 4. As already mentioned, the use of implicit time-stepping methods for convective problems is justified in applications not aiming to resolving features of the solution that require fine time resolution. Therefore we leave out of consideration issues, such as adaptive meshes, appropriate discretization using stabilization, CFL-condition.

8 | CONCLUDING REMARKS

Given their fundamental significance in many branches of science, solving time-dependent partial differential equations has been an important question for centuries and it remains an issue with high impact in many scientific computing applications. When handling such problems, the use of high order accurate implicit Runge–Kutta methods of Radau type can be numerically very efficient since the methods are strongly A-stable, enable use of large time-steps and can handle highly ill-conditioned matrices with a widespread spectrum. In this work, for the strongly A-stable IRK methods of Radau IIa type, applied to linear problems we show that thanks to the particular properties of the quadrature matrices A_q, originally derived in Reference 6 and holding true also for A_q^{-1}, the arising globally coupled linear systems on each time-step can be solved efficiently by a preconditioned iterative method with a stage-parallel (block-diagonal) preconditioner. Based on these properties, for the construction of the preconditioner we consider a high quality approximation of A_q^{-1}, namely, its lower-triangular part having real eigenvalues, and in this way fully avoid complex arithmetic. The implementation of the idea for two types of preconditioners, the analysis of the properties of the spectrum of the arising preconditioned nonsymmetric matrices (although currently not in its full generality for any number of stages) and the comprehensive performance tests are the major contributions of this article.

In this work, we consider only linear problems and constant timesteps. However, the approach allows to use automatic time step-size control for hybrid implicit-explicit methods enabling fine time resolution for a certain time interval and coarse time resolution in other parts of the time domain, switching to the considered here IRK method. As a suitable example we mention the simulation of the so-called glacial isostatic adjustment problem, where the uplift of the earth surface after melting of continental ice sheets is simulated for a period of about tens to hundreds of years and there the dynamic of the earth uplift is fast in the beginning but very slow in the larger of the time interval, see for instance Reference 71 for some details.

ACKNOWLEDGMENTS

The work of the second author (fully) and the third author (partly) is supported by Research Grant VR-2017-03749, "Mathematics and numerics in PDE-constrained optimization problems with state and control constraints," financed by the Swedish Research Council. The computations have been enabled by resources in project SNIC 2021/22-633 provided by the Swedish National Infrastructure for Computing (SNIC) at UPPMAX, partially funded by the Swedish Research Council through Grant agreement no. 2018-05973. Gratitude is extended to Peter Munch for his help in navigating deal.II and to Stefano Serra-Capizzano for the fruitful and insightful discussions. We are also indebted to the anonymous reviewers for their thorough reading and constructive suggestions towards improving the paper both structure- and contents-wise.

CONFLICT OF INTEREST STATEMENT

There are no conflicts of interest related to the study, presented in this paper.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Owe Axelsson https://orcid.org/0000-0001-8753-2429
Ivo Dravins https://orcid.org/0000-0002-0659-0596
Maya Neytcheva https://orcid.org/0000-0002-6719-4984

REFERENCES

1. Crank J, Nicolson PA. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. Math Proc Camb Philos Soc. 1947;43:50–67.
2. Dahlquist G. A special stability problem for linear multistep methods. BIT Numer Math. 1963;3:27–43.
3. Axelsson O. A class of A-stable methods. BIT Numer Math. 1969;9:185–99.
4. Ehle BL. High order A-stable methods for the numerical solution of systems of D.E.'s. BIT Numer Math. 1968;8:276–8.
5. Butcher JC. Implicit Runge-Kutta processes. Math Comput. 1964;18:50–64.
6. Axelsson O. Global integration of differential equations through Lobatto quadrature. BIT Numer Math. 1964;4:69–86.
7. Ehle BL. A-stable methods and Padé approximations to the exponential. SIAM J Math Anal. 1973;4:671–80.
8. Hairer E, Wanner G. Algebraically stable and implementable Runge-Kutta methods of higher order. SIAM J Numer Anal. 1981;18:1098–108.
9. Hairer E, Wanner G. Solving ordinary differential equations II: stiff and differential-algebraic problems. Berlin: Springer-Verlag; 1991.
10. Hairer E, Wanner G. Stiff differential equations solved by Radau methods. J Comput Appl Math. 1999;111:93–111.
11. Saad Y. Iterative methods for sparse linear systems. Philadelphia, PA: SIAM; 2003.
12. Hundsdorfer W, Verwer J. Numerical solution of time-dependent advection-diffusion-reaction equations. Berlin: Springer-Verlag; 2003.
13. Lambert JD. Numerical methods for ordinary differential systems. New York: Wiley; 1992.
14. Petzold L. Order results for implicit Runge-Kutta methods, applied to differential-algebraic systems. SIAM J Numer Anal. 1986;23:837–52.
15. Axelsson O, Blaheta R, Kohut R. Preconditioning methods for high-order strongly stable time integration methods with an application for a DAE problem. Numer Linear Algebra Appl. 2015;22:930–49.
16. Nørsett SP. Semi-explicit Runge-Kutta methods, report mathematics & computation. Trondheim, Norway: Department of Mathematics, University of Trondheim; 1974.
17. Zlatev Z. Modified diagonally implicit Runge-Kutta methods. SIAM J Sci Stat Comput. 1981;2:321–34.
18. van der Houwen PJ. Parallel step-by-step methods. Appl Numer Math. 1993;11:69–81.
19. van der Houwen PJ, Sommeijer BP. Iterated Runge-Kutta methods on parallel computers. SIAM J Sci Stat Comput. 1990;12:1000–28.
20. van der Houwen PJ, Sommeijer BP. Analysis of parallel diagonally implicit iteration of Runge-Kutta methods. Appl Numer Math. 1991;11(1993):169–88.
21. Axelsson O. On the efficiency of a class of A-stable methods. BIT Numer Math. 1974;14:279–87.
22. Butcher JC. Diagonally-implicit multi-stage integration methods. Appl Numer Math. 1993;11:347–63.
23. Hughes TJR, Hulbert GM. Space-time finite element methods for elastodynamics: formulations and error estimates. Comput Methods Appl Mech Eng. 1988;66:339–63.
24. Eriksson K, Johnson C. Adaptive finite element methods for parabolic problems I: a linear model problem. SIAM J Numer Anal. 1991;28:43–77.
25. Steinbach O, Yang H. 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. 2018;25:e2143.
26. Munch P, Dravins I, Kronbichler M, Neytcheva M. Stage-parallel fully implicit Runge-Kutta implementations with optimal multilevel preconditioners at the scaling limit. SIAM J Sci Comput. 2023:S71–S96.
27. Axelsson O, Neytcheva M. Numerical solution methods for implicit Runge-Kutta methods of arbitrarily high order. In: Frolkovič P, Mikula K, Ševčovič D, editors. Proceedings of the conference algoritmy. Bratislava: Slovak University of Technology in Bratislava; 2020. p. 11–20. http://www.iam.fmph.uniba.sk/amuc/ojs/index.php/algoritmy/article/view/1593/842
28. Axelsson O, Dravins I, Neytcheva M. Stage-parallel preconditioners for implicit Runge-Kutta methods of arbitrary high order: linear problems. Technical Report 2022-004. Uppsala: Department of Information Technology, Uppsala University; April 2022.
29. Hairer E, Wanner G. Construction of implicit Runge-Kutta methods. Solving ordinary differential equations II. Berlin: Springer; 1996. p. 71–90.
30. Hoffman W, de Swart J. Approximating Runge-Kutta matrices by triangular matrices. BIT Numer Math. 1997;37:346–54.
31. Pazner W, Persson P-O. Stage-parallel fully implicit Runge-Kutta solvers for discontinuous Galerkin fluid simulations. J Comput Phys. 2017;335:700–17.

32. Butcher JC. The numerical analysis of ordinary differential equations. Chichester: Wiley; 1987.
33. Axelsson O. A generalized conjugate gradient, least square method. Numer Math. 1987;51:209–27.
34. Jackson KR, Nørsett SP. The potential for parallelism in Runge-Kutta methods. Part I. RK formulas in standard form. SIAM J Numer Anal. 1995;32:49–82.
35. Alexander R. Diagonally implicit Runge-Kutta methods for stiff ODEs. SIAM J Numer Anal. 1977;14:1006–21.
36. van der Houwen P, de Swart JJB. Parallel linear system solver for Runge-Kutta methods. Adv Comput Math. 1997;7:157–81.
37. Gear CW, Xuhai X. Parallelism across time in ODEs. Appl Numer Math. 1993;11:45–68.
38. Lions J-L, Maday Y, Turinici G. A "parareal" in time discretization of PDEs. C R l'Académie Sci Ser I Math. 2001;332:661–8.
39. Butcher JC. On the implementation of implicit Runge-Kutta methods. BIT Numer Math. 1976;16:237–40.
40. Rana M, Howle V, Long K, Meek A, Milestone W. A new block preconditioner for implicit Runge-Kutta methods for parabolic PDE problems. SIAM J Sci Comput. 2021;43:S475–95. https://doi.org/10.1137/20M1349680
41. Southworth BS, Krzysik OA, Pazner W, De Sterck H. Fast solution of fully implicit Runge-Kutta and discontinuous Galerkin in time for numerical PDEs, part I: the linear setting. SIAM J Sci Comput. 2022;44(1):A416–43.
42. Jay LO, Braconnier T. A parallelizable preconditioner for the iterative solution of implicit Runge-Kutta type method. J Comput Appl Math. 1999;111:63–76.
43. Mardal K-A, Nilssen TK, Staff GA. Order optimal preconditioners for implicit Runge-Kutta schemes applied to parabolic PDEs. SIAM J Sci Comput. 2007;29:361–75.
44. Staff GA, Mardal K-A, Nilssen TK. Preconditioning of fully implicit Runge-Kutta schemes for parabolic PDEs. Model Identif Control. 2006;27:109–23.
45. Jiao X, Wang X, Chen Q. Optimal and low-memory near-optimal preconditioning of fully implicit Runge-Kutta schemes for parabolic PDEs. SIAM J Sci Comput. 2021;2021(43):A3527–51.
46. van der Houwen PJ, van der Veen WA. Waveform relaxation methods for implicit differential equations. Adv Comput Math. 1997;7:183–97.
47. Axelsson O, Lukáš D. Preconditioners for time-harmonic optimal control eddy-current problems. In: Lirkov I, Margenov S, editors. Large-scale scientific computing. Lecture Notes in Computer Science. Volume 10665. Cham: Springer; 2018. p. 47–54.
48. Axelsson O, Neytcheva M, Liang Z-Z. Parallel solution methods and preconditioners for evolution equations. Math Model Anal. 2018;23:287–308.
49. Sheen D, Sloan IH, Thomée V. A parallel method for time-discretization of parabolic problems based on contour integral representation and quadrature. Math Comput. 1999;69:177–95.
50. Sheen D, Sloan IH, Thomée V. A parallel method for time discretization of parabolic equations based on Laplace transformation and quadrature. IMA J Numer Anal. 2003;23:269–99.
51. Axelsson O, Verwer J. Boundary value techniques for initial value problems in ordinary differential equations. Math Comput. 1985;45:153–71.
52. Bellen A, Zennaro M. Parallel algorithms for initial-value problems for difference and differential equations. J Comput Appl Math. 1989;25:341–50.
53. Bolten M, Moser D, Spech R. A multigrid perspective on the parallel full approximation scheme in space and time. Numer Linear Algebra Appl. 2017;24:e2110.
54. Liu K, Liao X, Li Y. Parallel simulation of power systems transient stability based on implicit Runge–Kutta methods and W-transformation. Electr Power Compon Syst. 2017;47:2246–56.
55. Axelsson O, Blaheta R, Sysala S, Ahmad B. On the solution of high order stable time integration methods. Bound Value Probl. 2013;2013:108.
56. Abu-Labdeh R, MacLachlan SP, Farrell PE. Monolithic multigrid for implicit Runge-Kutta discretizations of incompressible fluid flow. J Comput Phys. 2023;478:111961.
57. Notay Y. An aggregation-based algebraic multigrid method. Electron Trans Numer Anal. 2010;37:123–46.
58. Vassilevski PS. Multilevel block factorization preconditioners. New York: Springer-Verlag; 2008.
59. Bickart TA. An efficient solution process for implicit Runge–Kutta methods. SIAM J Numer Anal. 1977;14:1022–7.
60. Axelsson O. Iterative solution methods. Cambridge: Cambridge University Press; 1994.
61. Gustafson KE, Rao DKM, Halmos PR. Numerical range: the field of values of linear operators and matrices. New York: Springer; 1996.
62. Horn RA, Johnson CR. Topics in matrix analysis. Cambridge: Cambridge University Press; 1991.
63. Fried I. Bounds on the spectral and maximum norms of the finite element stiffness, flexibility and mass matrices. Int J Solids Struct. 1973;9:1013–34.
64. Gustafsson I. Modified incomplete Cholesky (MIC) methods. In: Evans DJ, editor. Preconditioning methods: analysis and applications. Topics in Computational Mathematics. Volume 1. New York: Gordon & Breach; 1983. p. 265–93.
65. Axelsson O. A general incomplete block-matrix factorization method. Linear Algebra Appl. 1986;74:179–90.
66. Axelsson O, Vassilevski PS. Algebraic multilevel preconditioning methods. I. Numer Math. 1989;56:157–77.
67. Axelsson O, Neytcheva M. Algebraic multilevel iteration method for Stieltjes matrices. Numer Linear Algebra Appl. 1994;1:213–36.
68. Arndt D, Bangerth W, Blais B, Fehling M, Gassmöller R, Heister T, et al. The deal.II library, version 9.3. J Numer Math. 2021;29(3):171–86.
69. Balay S, Abhyankar S, Adams MF, Benson S, Brown J, Brune P, et al. PETSc/TAO users manual. ANL-21/39—Revision 3.16. Lemont, IL: Argonne National Laboratory; 2021.

70. Wang H, Dahle HK, Ewing RE, Espedal MS, Sharpley RC, Man S. An ELLAM scheme for advection-diffusion equations in two dimensions. SIAM J Sci Comput. 1999;20:2160–94.
71. Schmidt P, Lund B, Hieronymus C. Implementation of the glacial rebound prestress advection correction in general-purpose finite element analysis software: springs versus foundations. Comput Geosci. 2012;40:97–106.

How to cite this article: Axelsson O, Dravins I, Neytcheva M. Stage-parallel preconditioners for implicit Runge–Kutta methods of arbitrarily high order, linear problems. Numer Linear Algebra Appl. 2023;e2532. https://doi.org/10.1002/nla.2532