FAST SOLUTION OF FULLY IMPLICT RUNGE-KUTTA SYSTEMS IN NUMERICAL PDES*

BEN S. SOUTHWORTH †

Abstract.

1. Introduction.

1.1. Fully implicit Runge-Kutta. Consider the method-of-lines approach to the numerical solution of partial differential equations (PDEs), where we discretize in space and arrive at a system of ODEs in time,

$$M\mathbf{u}'(t) = \mathcal{N}(\mathbf{u}, t) \quad \text{in } (0, T], \quad \mathbf{u}(0) = \mathbf{u}_0, \tag{1}$$

where M is a mass matrix and $\mathcal{N} \in \mathbb{R}^{N\times N}$ is a discrete, time-dependent, nonlinear operator depending on t and $\mathbf{u}$ (including potential forcing terms). Then, consider time propagation using an s-stage Runge-Kutta scheme, characterized by the Butcher tableaux

$$\begin{array}{c|c} \mathbf{c}_0 & A_0 \\ \hline & \mathbf{b}_0^T \end{array},$$

with Runge-Kutta matrix $A_0 = (a_{ij})$, weight vector $\mathbf{b}_0^T = (b_1, \ldots, b_s)^T$, and quadrature nodes $\mathbf{c}_0 = (c_0, \ldots, c_s)$.

Runge-Kutta methods update the solution using a sum over stage vectors,

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \delta t \sum_{i=1}^{s} b_i \mathbf{k}_i,$$

$$M\mathbf{k}_i = \mathcal{N}\left(\mathbf{u}_n + \delta t \sum_{j=1}^{s} a_{ij}\mathbf{k}_j, t_n + \delta t c_i\right).$$

For nonlinear PDEs, $\mathcal{N}$ is linearized using, for example, a Newton or a Picard linearization of the underlying PDE. Let us denote this linearization $\mathcal{L} \in \mathbb{R}^{N\times N}$ (or, in the case of a linear PDE, let $\mathcal{L} := \mathcal{N}$). Expanding, solving for the stages $\mathbf{k}$ as each step in a nonlinear iteration, or as the update to $\mathbf{u}$ for a linear PDE, can then be expressed as a block linear system,

$$\left(\begin{bmatrix} M & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & M \end{bmatrix} - \delta t \begin{bmatrix} a_{11}\mathcal{L}_1 & \ldots & a_{1s}\mathcal{L}_1 \\ \vdots & \ddots & \vdots \\ a_{s1}\mathcal{L}_s & \ldots & a_{ss}\mathcal{L}_s \end{bmatrix}\right) \begin{bmatrix} \mathbf{k}_1 \\ \vdots \\ \mathbf{k}_s \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_s \end{bmatrix}. \tag{2}$$

The difficulty in fully implicit Runge-Kutta methods (which we will denote IRK) lies in solving the $Ns\times Ns$ block linear system in (2). This paper focuses on the parallel simulation of numerical PDEs, where N is typically very large, on the order of millions or billions, and $\mathcal{L}$ is highly ill-conditioned. In such cases, direct solution techniques

*This research was conducted ...

†Department of Applied Mathematics, University of Colorado at Boulder (ben.s.southworth@gmail.com).

This manuscript is for review purposes only.

to solve (2) are not a viable option, and fast, parallel iterative methods must be used. However, IRK methods are rarely employed in practice due to the difficulties of solving (2). Even for relatively simple parabolic PDEs where $\mathcal{L}$ is symmetric positive definite, (2) instead yields a large nonsymmetric system with significant block coupling. For nonsymmetric systems $\mathcal{L}$ that already have variable coupling, fast iterative methods are even less likely to yield acceptable performance in solving (2).

[TODO: Add citations] Although difficult to solve, there are a number of desirable properties of IRK schemes, particularly in terms of accuracy and stabillity ([TODO: why stability?]). In practice, people typically use diagonally implicit Runge-Kutta (DIRK) methods, where A_0 is lower triangular, or singly implicit Runge Kutta (SIRK) methods, where A_0 has exactly one positive real eigenvalue. For such schemes, the solution of (2) only requires s linear solves along the lines of $M - \delta t a_{ii}\mathcal{L}_i$. Unfortunately, SIRK and DIRK schemes suffer from stage-order one (stage-order two with one explicit stage), and for stiff nonlinear PDEs, the observed global order of accuracy in practice can be limited to $\approx \min\{p, q+1\}$, for integration order p and stage-order q. Thus, even a 6th-order DIRK method may only yield 2nd-order accuracy. For stiff problems, IRK methods can achieve high stage order (although not as high as the formal accuracy of the method), and thus result in high-order global accuracy. Furthermore, for less stiff problems, IRK methods can yield accuracy as high as order $2s$ for an s stage method, compared with a maximum of s or $s+1$ for SDIRK methods with reasonable stability properties [8, Section IV.6].

One simplification for using IRK methods in practice is to assume $\mathcal{L}_i = \mathcal{L}_j$ for all i, j, that is, $\mathcal{L}$ has no dependence on time. Such an assumption is natural for linear problems with no time-dependence in the spatial differential components, or when applying a simplified Newton method to nonlinear problems, where the Jacobian is only evaluated at one time-point per outer RK time step. Either case yields a simplified form of (2) that can be expressed in Kronecker product form,

$$(I \otimes M - \delta t A_0 \otimes \mathcal{L})\mathbf{k} = \mathbf{f}, \tag{3}$$

where $\mathcal{L}$ is a fixed real-valued spatial operator or Jacobian. Here, we revive some of the older Runge-Kutta work in the light of a changing computational landscape, developing a fast, parallel preconditioning technique for (3). The new method effectively requires s real-valued linear solves of matrices along the lines of $\eta M - \delta t\mathcal{L}$, for some $\eta > 1$, and is easily implemented using existing preconditioners and parallel software libraries. In contrast to other works that have considered the preconditioning of (3) via some form of splitting, the proposed algorithm here (i) is amenable to short-term Krylov recursion (conjugate gradient/MINRES) if $M - \mathcal{L}$ is, and (ii) only requires operating on systems of size N, an $s\times$ reduction in memory-usage of Krylov solvers compared with directly preconditioning the larger system. In addition, some of the results generalize to the weaker scenario of commuting spatial operators, $\mathcal{L}_i\mathcal{L}_j = \mathcal{L}_j\mathcal{L}_i$, which provides a first step towards the fast solution of time-dependent IRK.

[TODO: outline sections]

c:intro:prev}

1.2. Previous work. Many papers have considered the solution of (3) over the years. In 1976, Butcher [5] used the Jordan normal form, $A_0 = U_0 L_0 U_0^{-1}$, where L_0 is lower triangular with eigenvalues of A_0 on the diagonal, to transform (3) to the problem $(I\otimes M - \delta t L_0 \otimes \mathcal{L})(U_0^{-1}\otimes I)\mathbf{k} = (U_0 \otimes I)^{-1}\mathbf{f}$, where the inner operator is now block lower triangular. Such a transformation reduces the solution of an $Ns \times Ns$ system to s linear systems of size $N \times N$ in a block forward solve. The downside is that IRK schemes with greater accuracy and stability than DIRK schemes have

at most one real eigenvalue [6, 8]. Thus, for IRK methods such as Gauss or Radau integration, the original real-valued system is transformed into a set of smaller but primarily complex systems. There are various ways to handle the complex systems, but for numerial PDEs, the overhead in computational cost and implementation is typically too high to make the transformation a practical approach.

Published shortly after (and independently from) Butcher, Bickart proposed a similar way to invert (3) [2]. If we define $Q_s(x)$ as the characteristic polynomial of A_0, then the inverse of (3) can be computed via a specific set of matrix-vector multiplications in addition to the action of $Q_s(\mathcal{L})^{-1}$. In principle this is similar to Butcher's result, as one can invert $Q_s(\mathcal{L})$ by inverting each term in the factored polynomial, $(\mu_1 I - \mathcal{L})^{-1}$, $(\mu_2 I - \mathcal{L})^{-1}$, ..., for eigenvalues $\{\mu_i\}_{i=1}^s$ of A_0. Although Bickart's paper received less attention than Butcher's over time (currently 2.5× less citations), the polynomial form provides a more natural way to handle complex eigenvalues, particularly for numerical PDEs in the modern high-performance computing landscape, where direct LU inverses are rare and most linear systems are solved via preconditioning and/or Krylov methods. We present a similar result in Lemma 1 and use this to develop an effective preconditioning in Subsection 2.3.

Because significant research has been done on IRK methods, it is worth briefly reviewing some of the literature to put this work in context. In the field of time integration, SIRK and DIRK methods overcome the difficulty of solving IRK methods [1, 10]. In [11], it is shown that in considering RK methods with all real eigenvalues (wherein the Butcher transformation [5] remains real-valued), the best approximation to exponential is obtained by having all eigenvalues equal (SIRK methods [10]). Although SIRK methods offer some advantages over DIRK methods, they generally lack the favorable stability and accuracy of IRK methods [11]. ESIRK methods use one explicit stage and can offer stage-order two [4] (one higher than standard SIRK methods), which provide an improved practical option, but still lack the robustness of fully implicit methods.

{sec:solve}

2. Fast parallel solvers for IRK. The fully implicit RK stage system in (2) can be reformulated by pulling out an A_0^{-1} as [12]

{{eq:keq}}
$$(4)\quad \left(A_0^{-1} \otimes M - \delta t \begin{bmatrix} \mathcal{L}_1 & & \\ & \ddots & \\ & & \mathcal{L}_s \end{bmatrix}\right) (A_0 \otimes I) \begin{bmatrix} \mathbf{k}_1 \\ \vdots \\ \mathbf{k}_s \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_s \end{bmatrix}.$$

For ease of notation, let us scale both sides of the system by a block-diagonal mass matrix and, excusing the slight abuse of notation, let

$$\widehat{\mathcal{L}}_i := \delta t M^{-1} \mathcal{L}_i,$$

for $i = 1, .., s$. Note the time step δt for the given Runge-Kutta step is now included in $\widehat{\mathcal{L}}_i$. Now let α_{ij} denote the ij-element of A_0^{-1} (assuming A_0 is invertible). Then, solving (2) can be effectively reduced to inverting the operator

$$\mathcal{M}_s := A_0^{-1} \otimes I - \begin{bmatrix} \widehat{\mathcal{L}}_1 & & \\ & \ddots & \\ & & \widehat{\mathcal{L}}_s \end{bmatrix}$$

$$= \begin{bmatrix} \alpha_{11}I - \widehat{\mathcal{L}}_1 & \alpha_{12}I & ... & \alpha_{1s}I \\ \alpha_{21}I & \alpha_{22}I - \widehat{\mathcal{L}}_2 & & \alpha_{2s}I \\ \ddots & & \ddots & \vdots \\ \alpha_{s1}I & ... & \alpha_{s(s-1)}I & \alpha_{ss}I - \widehat{\mathcal{L}}_s \end{bmatrix}. \tag{5}$$

{{eq:k1}}

Note, there are a number of methods with one explicit stage preceded or followed by several fully implicit and coupled stages. In such cases, A_0 is not invertible, but the explicit stage can be eliminated from the system (by doing an explicit time step). The remaining operator can then be reformulated as above, and the inverse that must be applied takes the form of (5) but based on a principle submatrix of A_0.

{ec:solve:inv}

2.1. An inverse and update for commuting operators. This section introduces a result similar to Bickart's [2], but generalized to hold for commuting operators. If $\widehat{\mathcal{L}}_i = \widehat{\mathcal{L}}_j$ for all i, j, we show that the inverse of (5) can be expressed in terms of $P_s(\widehat{\mathcal{L}})^{-1}$, where $P_s(\widehat{\mathcal{L}})$ is the characteristic polynomial of A_0^{-1}. Note, we consider $\mathcal{M}_s$ as a matrix over the commutative ring of linear combinations of $\{I, \mathcal{L}\}$, and the determinant and adjugate referred to in Lemma 1 are defined over matrix-valued elements rather than scalars.

{lem:inv}

LEMMA 1. *Let α_{ij} denote the (i,j)th entry of A_0^{-1} and assume $\{\widehat{\mathcal{L}}_i\}_{i=1}^s$ are commuting operators. Define $\mathcal{M}_s$ as in (5). Let det$(\mathcal{M}_s)$ be the determinant of $\mathcal{M}_s$ (in this case a block-diagonal matrix) and let adj$(\mathcal{M}_s)$ be the adjugate of $\mathcal{M}_s$. Then, $\mathcal{M}_s$ is invertible if and only if det$(\mathcal{M}_s)$ is invertible, and*

$$\mathcal{M}_s^{-1} = \det(\mathcal{M}_s)^{-1}\mathrm{adj}(\mathcal{M}_s).$$

Now, suppose $\widehat{\mathcal{L}}_i = \widehat{\mathcal{L}}_j$ for all i, j, and let $P_s(x)$ be the characteristic polynomial of A_0^{-1}. Then,

$$\mathcal{M}_s^{-1} = \mathrm{diag}(P_s(\widehat{\mathcal{L}})^{-1})\mathrm{adj}(\mathcal{M}_s),$$

where "diag" indicates a block diagonal matrix, with diagonal blocks given by $P_s(\widehat{\mathcal{L}})^{-1}$.

Proof. Notice in (5) that if $\widehat{\mathcal{L}}_i$ and $\widehat{\mathcal{L}}_j$ commute for all i, j, then $\mathcal{M}_s$ is a matrix over the commutative ring of linear combinations of I and $\{\widehat{\mathcal{L}}_i\}$. Let adj$(\mathcal{M}_s)$ denote the matrix adjugate. A classical result in matrix analysis then tells us that

$$\mathrm{adj}(\mathcal{M}_s)\mathcal{M}_s = \mathcal{M}_s\mathrm{adj}(\mathcal{M}_s) = \det(\mathcal{M}_s)I.$$

Moreover, $\mathcal{M}_s$ is invertible if and only if if the determinant of $\mathcal{M}_s$ is invertible, in which case $\mathcal{M}_s^{-1} := \det(\mathcal{M}_s)^{-1}\mathrm{adj}(\mathcal{M}_s)$ [3, Theorem 2.19 & Corollary 2.21] For the case of time-independent operators ($\widehat{\mathcal{L}}_i = \widehat{\mathcal{L}}_j$), notice that $\mathcal{M}_s$ takes the form $A_0^{-1} - \widehat{\mathcal{L}}I$ over the commutative ring defined above. Analogous to a scalar matrix, the determinant of $A_0^{-1} - \widehat{\mathcal{L}}I$ is the characteristic polynomial of A_0^{-1} evaluated at $\widehat{\mathcal{L}}$. ∎

Returning to (4), we can express the direct solution for the set of all stage vectors $\mathbf{k} = [\mathbf{k}_1; ...; \mathbf{k}_s]$ as

$$\mathbf{k} := \det(\mathcal{M}_s)^{-1}(A_0^{-1} \otimes I)\mathrm{adj}(\mathcal{M}_s)\mathbf{f},$$

where $\mathbf{f} = [\mathbf{f}_1; ...; \mathbf{f}_s]$ (note that $A_0 \otimes I$ commutes with $\det(\mathcal{M}_s)^{-1}$). Excusing the slight abuse in notation, let $\det(\mathcal{M}_s)^{-1}$ now denote just the diagonal block (rather

than a block-diagonal matrix). The Runge-Kutta update is then given by

$$\begin{aligned}\mathbf{u}_{n+1} &= \mathbf{u}_n + \delta t \sum_{i=1}^{s} b_i \mathbf{k}_i \\ &= \mathbf{u}_n + \delta t \det(\mathcal{M}_s)^{-1} (\mathbf{b}_0^T A_0^{-1} \otimes I) \mathrm{adj}(\mathcal{M}_s) \mathbf{f}. \end{aligned} \tag{6}$$

The adjugate simply consists of linear combinations of I and $\widehat{\mathcal{L}}$, and an analytical form can be derived for an arbitrary $s \times s$ matrix, where $s \sim \mathcal{O}(1)$. Computing this operator analytically is easiest using a computer algebra program such as Mathematica. Applying its action will consist of some set of vector summations and matrix-vector multiplications. In particular, the diagonal elements of $\mathrm{adj}(\mathcal{M}_s)$ are monic polynomials in $\widehat{\mathcal{L}}$ of degree $s-1$ (or linear combinations of comparable degree if $\widehat{\mathcal{L}}_i \neq \widehat{\mathcal{L}}_j$) and off-diagonal terms are polynomials in $\widehat{\mathcal{L}}$ of degree $s-2$.

Returning to (6), we consider two cases. First, if a given Runge-Kutta scheme is stiffly accurate (for example, RadauIIA methods), then $\mathbf{b}_0^T A_0^{-1} = [0, ..., 0, 1]$. This yields the nice simplification that computing the update in (6) only requires applying the last row of $\mathrm{adj}(\mathcal{M}_s)$ to $\mathbf{f}$ (in a dot-product sense) and applying $\det(\mathcal{M}_s)^{-1}$ to the result. From the discussion above regarding the adjugate structure, applying the last row of $\mathrm{adj}(\mathcal{M}_s)$ requires $(s-2)(s-1)+(s-1) = (s-1)^2$ matrix-vector multiplications. Because this only happens once, followed by the linear solve(s), these multiplications are typically of relatively marginal cost.

In the more general case of non stiffly accurate (for example, Gauss methods), one can obtain an analytical form for $(\mathbf{b}_0^T A_0^{-1} \otimes I)\mathrm{adj}(\mathcal{M}_s)$. Each element in this matrix consists of polynomials in $\widehat{\mathcal{L}}$ of degree $s-1$ (although typically not monic). Compared with stiffly accurate schemes, this now requires $(s-1)s$ matrix-vector multiplications, which is $s-1$ more than for stiffly accurate schemes, but still typically of marginal overall computational cost.

{sec:solve:stab}

2.2. Stability and the method of lines.

- A necessary condition for stable time-propagation is that eigenvalues of $\widehat{\mathcal{L}}$ lie in the region of stability for the given RK scheme. For virtually all implicit RK schemes, this means that eigenvalues of $\widehat{\mathcal{L}}$ must have negative real part, or real part $\gg 0$. It is rare for operators representing a differential discretization to be split with a set of negative definite eigenvalues and a set of eigenvalues with real part $\gg 0$, so for our purposes assume $\widehat{\mathcal{L}}$ has eigenvalues with negative real part.
- Just eigenvalues is not strong enough for our analysis or stability. Need to look into literature. Would be nice if it was related to numerical range/field of values, because that is what our analysis is based on. Ideally, something along the lines of $\widehat{\mathcal{L}}$ has a non-positive field of values for stability?

{sec:solve:prec}

2.3. Preconditioning by conjugate pairs. Following the discussion and algorithm developed in Subsection 2.1, the key outstanding point is inverting $\det(\mathcal{M}_s)^{-1}$. Moving forward, we restrict our attention to the case $\widehat{\mathcal{L}}_i = \widehat{\mathcal{L}}_j$ for all i, j, in which case $\det(\mathcal{M}_s)^{-1} = P_s(\widehat{\mathcal{L}})^{-1}$, where $P_s(x)$ is the characteristic polynomial of A_0^{-1} (see Lemma 1).

In contrast to much of the original work on solving IRK systems, where LU factorizations were the dominant cost and system sizes relatively small, explicitly forming and inverting $P_s(\widehat{\mathcal{L}})$ for numerical PDEs is typically not a viable option in high-performance simulation on modern computing architectures. Instead, by computing

the eigenvalues $\{\lambda_i\}$ of A_0^{-1}, we can express $P_s(\widehat{\mathcal{L}})$ in a factored form,

$$P_s(\widehat{\mathcal{L}}) = \prod_{i=1}^{s} (\lambda_i I - \widehat{\mathcal{L}}), \tag{7}$$

{eq:fac}

and its inverse can then be computed by successive applications of $(\lambda_i I - \widehat{\mathcal{L}})^{-1}$, for $i = 1, ..., s$. As discussed previously, eigenvalues of A_0 and A_0^{-1} will often be complex, making the inverse of individual factors $(\lambda_i I - \widehat{\mathcal{L}})^{-1}$ more difficult and often impractical.

Here, we propose combining pairs of conjugate eigenvalues into quadratic polynomials that we must precondition, which take the form

$$\begin{aligned} \mathcal{Q}_\eta :&= ((\eta + i\beta)I - \widehat{\mathcal{L}})((\eta - i\beta)I - \widehat{\mathcal{L}}) \\ &= (\eta^2 + \beta^2)I - 2\eta\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^2 = (\eta I - \widehat{\mathcal{L}})^2 + \beta^2 I. \end{aligned} \tag{8}$$

In practice, we typically do not want to directly form or precondition a quadratic operator like (8), due to (i) the overhead cost of large parallel matrix multiplication, and (ii) the fact that many fast parallel methods such as multigrid are not well-suited for solving a polynomial in $\widehat{\mathcal{L}}$. The point of (8) is that by considering conjugate pairs of eigenvalues, the resulting operator is real-valued. Thus, consider preconditioning (8) with the inverse of the real-valued quadratic polynomial, $(\eta I - \widehat{\mathcal{L}})^2$, dropping the $+\beta^2 I$ term. Expanding, the preconditioned operator then takes the form

$$\begin{aligned} \mathcal{P}_\eta &:= (\eta I - \widehat{\mathcal{L}})^{-2} \left[(\eta^2 + \beta^2)I - 2\eta\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^2\right] \\ &= I + \beta^2(\eta I - \widehat{\mathcal{L}})^{-2} = I + \frac{\beta^2}{\eta^2}\left(I - \tfrac{1}{\eta}\widehat{\mathcal{L}}\right)^{-2}. \end{aligned} \tag{9}$$

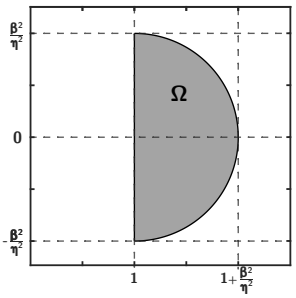
It turns out, by reasonable assumptions on $\widehat{\mathcal{L}}$ that yield a stable time-propagation scheme [TODO: reference?], the preconditioned operator in (9) is very well-conditioned. Theorem 3 analyzes the field-of-values of $\mathcal{P}_\eta$, denoted $W(\mathcal{P}_\eta)$, as a measure of the preconditioning, and Corollary 4 extends this preconditioning to prove fast convergence of GMRES.

Remark 2 (Convergence independent of problem size). It is worth pointing out the key cancellation in (9), where the preconditioned operator has no terms directly involving $\widehat{\mathcal{L}}$ or $\widehat{\mathcal{L}}^{-1}$ (why $(I - \frac{1}{\eta}\widehat{\mathcal{L}})^{-2}$ is different will be clear in the proof of Theorem 3). Because of this, the conditioning of $\mathcal{P}_\eta$ is effectively independent of the conditioning of $\widehat{\mathcal{L}}$, which ensures convergence that is independent of problem size. This is in contrast to most standard matrix-splitting or block-preconditioning techniques, where the preconditioned operator has some dependence on $\widehat{\mathcal{L}}$, which leads to convergence dependent on the conditioning of $\widehat{\mathcal{L}}$ (and, thus indrectly, problem size).

{th:fov}

THEOREM 3 (Preconditioned field of values). *Assume that $\eta > 0$ and the symmetric part of $\widehat{\mathcal{L}}$ satisfies $(\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^T)/2 \leq 0$. Let $\mathcal{P}_\eta$ denote the preconditioned operator, where $((\eta + i\beta)I - \widehat{\mathcal{L}})((\eta - i\beta)I - \widehat{\mathcal{L}})$ is preconditioned with $(\eta I - \widehat{\mathcal{L}})^{-2}$. Then $W(\mathcal{P}_\eta)$ is bounded by Ω as shown in Figure 1.*

Proof. Let $\sigma(B)$ denote the spectrum of operator B, $\sigma_{\min}(B)$ and $\sigma_{\max}(B)$ the minimum and maximum eigenvalues, and $\rho(B)$ the spectral radius. Also, define the symmetric/skew-symmetric splitting $B = B_s + B_k$, where $B_s := (B + B^T)/2$ and $B_k := (B - B^T)/2$, and the numerical radius as $r(B) = \sup\{|\lambda| : \lambda \in W(B)\}$. Recall the following properties of $W(B)$ [7]:



{fig:bound} FIG. 1.

1. $W(B) \subset [\sigma_{\min}(B_s), \sigma_{\max}(B_s)] \times [-\rho(B_k)\mathrm{i}, \rho(B_k)\mathrm{i}]$.
2. $\sigma(B) \subset W(B)$.
3. If B is invertible and $B_s \leq 0$ in the symmetric negative semi-definite sense, then the symmetric part of B^{-1} is also negative semi-definite.
4. $r(B) \leq \|B\|_2$.
5. $W(I + B) = 1 + W(B)$.

Note that an exact inverse yields $\mathcal{P}_\eta = I$, with spectrum and field-of-values given by $\sigma(\mathcal{P}_S) = W(\mathcal{P}_S) = \{1\}$. Appealing to (9) and the final property stated above, $W(\mathcal{P}_\eta) = 1 + \frac{\beta^2}{\eta^2} W(E)$, for error term $E := (I - \frac{1}{\eta}\widehat{\mathcal{L}})^{-2}$ and real-valued constant $\beta^2/\eta^2 > 0$. Next we will bound $W(E)$ in the complex plane.

Assume that $\eta > 0$ and the symmetric part of $\widehat{\mathcal{L}}$ satisfies $(\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^T)/2 \leq 0$. It follows that the real part of eigenvalues of $\widehat{\mathcal{L}}$ are non-positive and, thus, $(I - \frac{1}{\eta}\widehat{\mathcal{L}})$ cannot have a zero eigenvalue and must be invertible. Furthermore, it also follows that the symmetric part of $(I - \frac{1}{\eta}\widehat{\mathcal{L}})$ is symmetric positive definite and thus the symmetric part of $(I - \frac{1}{\eta}\widehat{\mathcal{L}})^{-2}$ is as well. This yields a lower bound of zero on the real-axis for $W(E)$, that is, $\mathrm{Re}(W(E)) > 0$.

Now, note that by the assumption $(\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^T)/2 \leq 0$, we have

$$\frac{\left\langle (I - \frac{1}{\eta}\widehat{\mathcal{L}})\mathbf{x}, (I - \frac{1}{\eta}\widehat{\mathcal{L}})\mathbf{x} \right\rangle}{\langle \mathbf{x}, \mathbf{x} \rangle} = 1 - \frac{\langle (\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^T)\mathbf{x}, \mathbf{x} \rangle}{\eta \langle \mathbf{x}, \mathbf{x} \rangle} + \frac{\langle \frac{1}{\eta^2}\widehat{\mathcal{L}}^T\widehat{\mathcal{L}}\mathbf{x}, \mathbf{x} \rangle}{\langle \mathbf{x}, \mathbf{x} \rangle} \geq 1 \tag{10}$$

{eq:norm1}

for all $\mathbf{x} \neq \mathbf{0}$. Then,

$$\begin{aligned} \|(I - \tfrac{1}{\eta}\widehat{\mathcal{L}})^{-2}\| \leq \|(I - \tfrac{1}{\eta}\widehat{\mathcal{L}})^{-1}\|^2 &= \sup_{\mathbf{x} \neq \mathbf{0}} \frac{\left\langle (I - \frac{1}{\eta}\widehat{\mathcal{L}})^{-1}\mathbf{x}, (I - \frac{1}{\eta}\widehat{\mathcal{L}})^{-1}\mathbf{x} \right\rangle}{\langle \mathbf{x}, \mathbf{x} \rangle} \\ &= \sup_{\mathbf{y} \neq \mathbf{0}} \frac{\langle \mathbf{y}, \mathbf{y} \rangle}{\left\langle (I - \frac{1}{\eta}\widehat{\mathcal{L}})\mathbf{y}, (I - \frac{1}{\eta}\widehat{\mathcal{L}})\mathbf{y} \right\rangle} \leq 1. \end{aligned}$$

This yields a bound on the numerical radius $r(E) = r((I - \frac{1}{\eta}\widehat{\mathcal{L}})^{-2}) \leq \|(I - \frac{1}{\eta}\widehat{\mathcal{L}})^{-2}\| \leq 1$. Combining with $\mathrm{Re}(W(E)) > 0$, the field of values of the error term, $W(E)$, is contained in the positive half of the unit circle in the complex plane, which completes the proof. ∎

{cor:gmres}

COROLLARY 4 (GMRES convergence bounds). *Let π_k denote the set of consistent polynomials of degree k. Then the ideal GMRES bound (an upper bound in*

operator norm of worst-case convergence) on convergence after k iterations applied to the preconditioned operator $\mathcal{P}_\eta$ (9) is bounded by

$$\min_{p\in\pi_k} \|p(\mathcal{P}_\eta)\| \leq 2\left(\frac{\beta^2/\eta^2}{2+\beta^2/\eta^2}\right)^k.$$

Proof. For operator B, let $\nu(B)$ denote the distance of $W(B)$ from the origin and define $\cos(\beta) := \nu(B)/r(B)$. In [9, Lemma 3.2], it is proven that worst-case convergence of GMRES applied to operator B is bounded by

$$\min_{p\in\pi_k} \|p(B)\| \leq 2\left(\frac{1-\cos\beta}{1+\cos\beta}\right)^k. \tag{11}$$

For $B = \mathcal{P}_\eta$, we have $\nu(\mathcal{P}_\eta) = 1$ and $r(\mathcal{P}_\eta) \leq 1+\beta^2/\eta^2$. Plugging into (11) completes the proof.[1] ∎

{cor:cg}

Corollary 5 (CG convergence bounds). *Define $\mathcal{Q}_\eta$ as in (8) and $\mathcal{P}_\eta$ as in (9), and assume $(\eta I - \widehat{\mathcal{L}})$ is SPD. Then, the error $\mathbf{e}_k$ in the $\mathcal{Q}_\eta$-norm after k iterations of preconditioned conjugate gradient is bounded by*

$$\frac{\|\mathbf{e}_k\|_{\mathcal{Q}_\eta}}{\|\mathbf{e}_0\|_{\mathcal{Q}_\eta}} \leq 2\left(\frac{\sqrt{1+\beta^2/\eta^2}-1}{\sqrt{1+\beta^2/\eta^2}+1}\right)^k.$$

Proof. Note that if $(\eta I - \widehat{\mathcal{L}})$ is SPD, then $\mathcal{Q}_\eta$ is also SPD. Then, recall that for conjugate gradient applied to SPD matrix A, error is bounded via

$$\frac{\|\mathbf{e}_k\|_A}{\|\mathbf{e}_0\|_A} \leq 2\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^k, \tag{12}$$

where $\kappa(A)$ denotes the condition number of A. For preconditioned CG with SPD preconditioner B^{-1} and Cholesky decomposition $B = R^T R$, PCG applied to $A\mathbf{x} = \mathbf{b}$ is equivalent to applying CG to the modified SPD system $(R^{-T}AR^{-1})R\mathbf{x} = R^{-T}\mathbf{b}$. The condition number is then given by

$$\begin{aligned}\kappa(R^{-T}AR^{-1}) &= \lambda_{max}(R^{-T}AR^{-1})/\lambda_{min}(R^{-T}AR^{-1})\\ &= \lambda_{max}(R^{-1}R^{-T}A)/\lambda_{min}(R^{-1}R^{-T}A)\\ &= \lambda_{max}(B^{-1}A)/\lambda_{min}(B^{-1}A).\end{aligned}$$

for largest and smallest eigenvalues λ_{max} and λ_{min}, respectively. Then, recall that eigenvalues of an operator are contained in the field-of-values, and from Theorem 3. This yields $\lambda_{max}(\mathcal{P}_\eta)/\lambda_{min}(\mathcal{P}_\eta) \leq 1+\beta^2/\eta^2$, and by monotonicity of (12) in κ this completes the proof. ∎

Interestingly, in dropping the $\beta^2 I$ term from $\mathcal{Q}_\eta$ in the preconditioner, we are actually preconditioning $\mathcal{Q}_\eta = (\eta^2+\beta^2)I - 2\eta\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^2$ with a worse-conditioned operator, since $\beta^2 I$ shifts the spectrum and field of values positive away from the origin. Nevertheless, this preconditioning is quite effective – Table 1 provides the values of η, β^2/η^2, and the CG and GMRES bounds from Corollary 4 and Corollary 5 for Gauss, RadauIIA, and LobattoIIIC integration.

[1] Note, classical GMRES convergence results based on $\lambda_{\min}((\mathcal{P}_\eta + \mathcal{P}_\eta^T)/2)$ and $\lambda_{\max}(\mathcal{P}_\eta^T\mathcal{P}_\eta)$ can also be applied, yielding the bound $\left(\frac{\beta^2/\eta^2}{1+\beta^2/\eta^2}\right)^{k/2}$, $2-4\times$ less tight than Theorem 3.

	Stages	2	3		4		5		
Gauss	η	3.0	4.64	3.68	4.21	5.79	7.29	4.65	6.70
	β^2/η^2	0.33	0	0.91	1.59	0.09	0	2.36	0.27
	CG	0.072	0	0.160	0.234	0.022	0	0.294	0.060
	GMRES	0.143	0	0.313	0.444	0.043	0	0.541	0.119
RadauIIA	η	2.0	3.64	2.68	3.21	4.79	6.29	3.66	5.70
	β^2/η^2	0.50	0	1.29	2.21	0.11	0	3.20	0.32
	CG	0.101	0	0.205	0.283	0.025	0	0.344	0.069
	GMRES	0.2	0	0.393	0.525	0.051	0	0.616	0.1368
LobattoIIIC	η	1.0	2.63	1.69	2.22	3.78	...		
	β^2/η^2	1	0	2.21	3.51	0.13	...		
	CG	0.172	0	0.284	0.360	0.031	...		
	GMRES	0.333	0	0.525	0.637	0.063	...		

TABLE 1

η, β^2/η^2, and the convergence factors in Corollary 4 and Corollary 5 for GMRES and CG, respectively (without the leading constants of 2) for Gauss, RadauIIA, and LobattoIIIC integration, with 2–5 stages. Each column within a given set of stage columns corresponds to a single eigenvalue of A_0^{-1}. For odd numbers of stages, one eigenvalue is real, corresponding to the column where $\beta^2/\eta^2 = 0/\eta^2 = 0$. [TODO: LobattoIIC s=5]

{tab:beta}

2.3.1. Inexact preconditioning. In practice, fully converging $(\eta I - \widehat{\mathcal{L}})^{-1}$ as a preconditioner is not desirable – even if a Krylov method converges rapidly, if each iteration requires a full linear solve, the resulting method remains moderately expensive. Here, we propose applying a Krylov method to $(\eta^2 + \beta^2)I - 2\eta\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^2$ by computing the operator's action (that is, not fully constructing it), and preconditioning each Krylov iteration with *two* applications of a sparse parallel preconditioner for $(\eta I - \widehat{\mathcal{L}})$, representing the action of $(\eta I - \widehat{\mathcal{L}})^{-2}$.

To motive this heuristically, suppose $(\eta I - \widehat{\mathcal{L}}) = (\eta I - B) + N$, where $(\eta I - B)^{-1}$ corresponds to the action of a preconditioner and N the error terms that is not capture. For a good preconditioner, we expect $(\eta I - B)^{-1}(\eta I - \widehat{\mathcal{L}}) = I + (\eta I - B)^{-1}N$ to be well-conditioned and, thus, $(\eta I - B)^{-1}N$ to be small in some sense. Then,

$$(\eta I - B)^{-2}\mathcal{Q}_\eta = (\eta I - B)^{-2}(\eta I - \widehat{\mathcal{L}})^2 + \frac{\beta^2}{\eta^2}(\eta I - B)^{-2}.$$

[TODO: Fix this section] Note the terms $B^{-1}N + B^{-2}NB + B^{-2}N^{-2}$ exactly correspond to error propagation of one fixed-point iteration to precondition $(\eta I - \widehat{\mathcal{L}})^2$ with B^{-2}, which we expect to be small if B^{-1} is an effective preconditioner for $(\eta I - \widehat{\mathcal{L}})^2$. Thus, there is a sense of commutativity of preconditioning, where two applications of a preconditioner for $(\eta I - \widehat{\mathcal{L}})$ will serve as an effective preconditioning for $((\eta^2+\beta^2)I - 2\eta\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^2)$. This is confirmed in practice for several different examples in Section 4.

It also must be mentioned that if an IRK method is A-stable, irreducible, and A_0 is invertible (which includes Gauss, RadauIIA, and LobattoIIIC integration), then $\eta > 0$ [8]. From the preconditioning perspective this is important, as if $\eta > 0$ and $\widehat{\mathcal{L}}$ has negative real part, then $(\eta I - \widehat{\mathcal{L}})$ has positive real part bounded nicely away from zero, and is thus invertible and amenable to preconditioning.

2.3.2. Mass matrices. Recall in the finite element context where mass matrices are involved, we defined $\widehat{\mathcal{L}} := \delta t M^{-1}\mathcal{L}$. For a given conjugate pair of eigenvalues, the quadratic polynomial can be expressed as

$$M^{-1}((\eta+i\beta)M - \delta t\mathcal{L})M^{-1}((\eta-i\beta)M - \delta t\widehat{\mathcal{L}}). \tag{13}$$

In this context, it is best to first scale both sides of the linear system by M. This halves the number of times M^{-1} must applied for each Krylov iteration, and if M and $\mathcal{L}$ are Hermitian, the resulting quadratic system is SPD and can be solved using preconditioned conjugate gradient or MINRES, preconditioned with one application of a preconditioner, the action of M, and a second application of the preconditioner.

3. Implementation details: The time-independent case. The basic idea is that we need to do the update

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \delta t \det(\mathcal{M}_s)^{-1}\left(\boldsymbol{b}_0^\top A_0^{-1}\otimes I\right)\operatorname{adj}(\mathcal{M}_s)\boldsymbol{f}, \tag{14}$$

where we have

- $m \in \mathbb{N}$: The dimension of the spatial problem
- $\boldsymbol{f} = (\boldsymbol{f}_1, \dots, \boldsymbol{f}_s)^\top$, with $\boldsymbol{f}_j \in \mathbb{R}^m$
- $\tilde{\boldsymbol{b}}_0^\top := \boldsymbol{b}_0^\top A_0^{-1} \in \mathbb{R}^{1\times s}$ (this is a row vector, not that this distinction really matters)

Now we break (14) into two steps:

update_step1

1. Step 1:Compute

$$\boldsymbol{z} = \left(\hat{\boldsymbol{b}}_0^\top \otimes I\right)\operatorname{adj}(\mathcal{M}_s)\boldsymbol{f} \in \mathbb{R}^m \tag{15}$$

update_step2

2. Step 2: Solve

$$\det(\mathcal{M}_s)\boldsymbol{y} = \boldsymbol{z}, \tag{16}$$

then update per (14), $\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \delta t\boldsymbol{y}$

3.1. On Step 1. The adjugate of $\mathcal{M}_s$ is a matrix defined over degree $s-1$ polynomials in $\mathcal{L} \in \mathbb{R}^{m\times m}$. More specifically, we write it as

$$\operatorname{adj}(\mathcal{M}_s) = \begin{bmatrix} Q_{11}(\mathcal{L}) & \cdots & Q_{1s}(\mathcal{L}) \\ \vdots & & \vdots \\ Q_{s1}(\mathcal{L}) & \cdots & Q_{ss}(\mathcal{L}) \end{bmatrix} \in \mathbb{R}^{ms\times ms}, \quad Q_{ij}(\mathcal{L}) = \sum_{k=0}^{s-1}\hat{q}_{ijk}\mathcal{L}^k \in \mathbb{R}^{m\times m}. \tag{17}$$

Now, the Kronecker product appearing in front of this matrix simply takes inner products over its columns to give a block rectangular matrix whose elements are therefore polynomials in $\mathcal{L}$ of degree $s-1$:

$$\left(\boldsymbol{b}_0^\top A_0^{-1}\otimes I\right)\operatorname{adj}(\mathcal{M}_s) = \left[X_1(\mathcal{L}), \cdots, X_s(\mathcal{L})\right], \tag{18}$$

where

$$X_j(\mathcal{L}) = \sum_{k=0}^{s-1}\hat{x}_{jk}\mathcal{L}^k \in \mathbb{R}^{m\times m}, \quad \hat{x}_{jk} = \sum_{\ell=1}^{s}\left(\tilde{\boldsymbol{b}}_0^\top\right)_\ell \hat{q}_{\ell jk}. \tag{19}$$

And, so, finally, the vector in (15) can be written as the sum

$$\boldsymbol{z} = \sum_{i=1}^{s} X_i(\mathcal{L})\boldsymbol{f}_i. \tag{20}$$

The main task here is going to be computing the action of the degree $s-1$ polynomials $X_i(\mathcal{L})$ of the components of $\boldsymbol{f}$; note that we can easily compute the sets of coefficients $\{x_{jk}\}_{(j,k)=(1,0)}^{(s,s-1)}$. I think the most efficient way to compute the action of this polynomial is with a Horner-like scheme which is a well-known technique for evaluating scalar polynomials (see https://en.wikipedia.org/wiki/Horner%27s_method). Basically, we can compute the action of the nth degree polynomial $P_n(\mathcal{L})$ on a vector using: n `MATVECs` with $\mathcal{L}$, $n+1$ `AXPYs` ($\boldsymbol{x} \leftarrow \alpha\boldsymbol{y} + \beta\boldsymbol{z}$), n `copies` (n lots of copying values from one vector to another, $n \times [\boldsymbol{x} \leftarrow \boldsymbol{y}]$), and one intermediate/temporary vector. So the main cost in computing (20) is $s(s-1)$ `MATVECs` with $\mathcal{L}$.

{sec:numerics}

4. Numerical results.

5. The linear time-dependent setting. Are there any problems with time-dependent components in the differential operator that are linear? Since we are integrating in time, I am thinking this will become nonlinear inherently. If not, it is possible something like this could work. We want to solve $A\mathbf{k} = \mathbf{f}$, and then sum over stages $\mathbf{k}$ using the block multiplication $(A_0^{-1}\mathbf{b}_0 \otimes I)\mathbf{k} = (A_0^{-1}\mathbf{b}_0 \otimes I)A^{-1}\mathbf{f}$.

{sec:nonlinear}

6. The nonlinear setting. For linear problems, one of the main benefits of the approach developed in Section 2 is that the algorithm can be applied in some sense on the solution level rather than solving for all stages simultaneously and then updating the solution by summing over stages. This is particularly beneficial from a memory perspective, being able to achieve very high order accuracy while only storing the solution and an auxilliary vector, and also allows for the use of CG and MINRES, when applicable, to the underlying system.

Unfortunately, the time-dependent and nonlinear case is more complicated. Consider the simplest cast of a time-independent nonlinear problem and simplified Newton method, which only applies the Jacobian based on the current solution. Each nonlinear iteration requires the action of the nonlinear operator to compute a residual, and this action is implicitly defined by individual stage vectors. The linear algorithm developed in Section 2 solves for the summation over stage vectors, and individual stages cannot be extracted. Without each stage vector, the action of the nonlinear operator cannot be computed, and even the simplified Newton method cannot be applied. Similar difficulties apply for full Newton and Picard iterations for operators with and without time-dependent differential components. Reformulating the algorithm from Section 2 to store stage vectors also ends up being impractical – applying $\det\mathcal{M}_s^{-1}$ requires the solution of s linear systems. To store each stage vector requires applying $\det\mathcal{M}_s^{-1}$ to each one. This requires s^2 linear solves, which is too expensive to be appealling in practice.

{sec:nonlinear:simp}

6.1. Simplified Newton.

It appears that all stage vectors must be stored to perform an implicit Runge-Kutta iteration using standard nonlinear methods. To develop an efficient approach to solving the linearized systems, let us consider developing a similar method as in Section 2, but which stores all vectors and only requires s linear solves. Suppose $\mathcal{L}_i = \mathcal{L}_j$ for all i, j (as in a simplified Newton method). Returning to (4), the stage updates are defined as the solution of the linear system

$$\left(A_0^{-1} \otimes I - I \otimes \widehat{\mathcal{L}}\right)(A_0 \otimes I)\mathbf{k} = \mathbf{f}. \tag{21}$$

{eq:keq2}

Now, let $A_0^{-1} = Q_0 R_0 Q_0^T$ be the real Schur decomposition of A_0^{-1}, where Q_0 is real-valued and orthogonal, and R_0 is a block upper triangular matrix, where each block corresponds to an eigenvalue (pair) of A_0^{-1}. Real-valued eigenvalues have block size one, and complex eigenvalues $\eta \pm i\beta$ are in 2×2 blocks, $\begin{bmatrix} \eta & \beta \\ -\beta & \eta \end{bmatrix}$. Pulling out a $Q_0 \otimes I$ and $Q_0^T \otimes I$ from the left and right of (21) yields the equivalent linear system

$$\left(R_0 \otimes I - I \otimes \widehat{\mathcal{L}}\right)(R_0^{-1}Q_0^T \otimes I)\mathbf{k} = (Q_0^T \otimes I)\mathbf{f}. \tag{22}$$

The left-most matrix is now a block upper triangular matrix, which can be solved using a block backward solve, and requires inverting each diagonal block. Diagonal blocks corresponding to real-valued eigenvalues η take the form $(\eta I - \widehat{\mathcal{L}})$, and are amenable to standard preconditioning texhniques. 2×2 diagonal blocks corresponding to complex eigenvalues take the form

$$\begin{bmatrix} \eta I - \widehat{\mathcal{L}} & \beta I \\ -\beta I & \eta I - \widehat{\mathcal{L}} \end{bmatrix}. \tag{23}$$

It turns out analogous ideas as used for the adjugate formulation can be applied to the 2×2 operator in (23). Note that the Schur complement of (23) is given by

$$S := \eta I - \widehat{\mathcal{L}} + \beta^2(\eta I - \widehat{\mathcal{L}})^{-1},$$

and consider a block lower triangular preconditioner for (23) given by

$$L_P := \begin{bmatrix} \eta I - \widehat{\mathcal{L}} & \mathbf{0} \\ -\beta I & \widehat{S} \end{bmatrix}^{-1}. \tag{24}$$

When applying GMRES to block 2×2 operators preconditioned with a lower (or upper) triangular preconditioner (analogous to (24)), convergence is exactly defined by convergence of GMRES applied to the preconditioned Schur complement, $\widehat{S}^{-1}S$ [TODO: cite]. If $\widehat{S} = S$ is exact, exact convergence on the larger 2×2 system is guaranteed in two iterations (or one iteration with a block LDU).

Similar to Subsection 2.3, suppose we precondition S by dropping terms corresponding to the complex part of the eigenvalue, β^2, yielding the preconditioned operator

$$\begin{aligned} (\eta I - \widehat{\mathcal{L}})^{-1}S &:= I + \beta^2(\eta I - \widehat{\mathcal{L}})^{-2} \\ &= I + \frac{\beta^2}{\eta^2}(I - \tfrac{1}{\eta}\widehat{\mathcal{L}})^{-2}. \end{aligned} \tag{25}$$

This is exactly the form we arrived at when analyzing preconditioning using the adjugate and determinant in Section 2 and Theorem 3. Indeed, because GMRES convergence of the preconditioned 2×2 operator is defined by GMRES convergence on the preconditioned Schur complement, the GMRES bounds presented in Table 1 apply here as well. Similar to Section 2, we do not want to solve $(\eta I - \widehat{\mathcal{L}})^{-1}$ exactly each iteration. It is well-known in the block-preconditioning community that a few iterations of an effective preconditioner typically yields convergence on the 2×2 operator just as fast as if performing direct solves in the preconditioning.

Thus, we have an algorithm that is similar in essence to that of Section 2, but is able to store all stage vectors and is compatible with a simplified Newton iteration.

Note, for linear problems the algorithm from Section 2 is still preferable because the nonlinear method presented here requires more memory to store stage vectors, is not amenable to the use of CG/MINRES, and the Krylov method of choice must store two stages instead of one for each vector in the Krylov space. On the bright side, the method based on a real Schur decomposition does give insight into a preconditioning for the fully nonlinear and time-dependent setting, which is discussed in the next section.

Remark 6 (Closed form Inverse). One might note that the two Schur complements of (23) are the same, and

$$S^{-1} = \mathcal{Q}_\eta^{-1}(\eta I - \widehat{\mathcal{L}}),$$

with $\mathcal{Q}_\eta$ as in (8). Using this, one can get a simple closed form for the inverse of (8) based on $\mathcal{Q}_\eta^{-1}$. However, consistent with the algorithm developed in Section 2, applying the 2×2 inverse requires applying $\mathcal{Q}_\eta^{-1}$ to both stage vectors, which doubles the number of linear solves.

Remark 7 (A similar approach for A_0). This is not the first work to consider a real Schur decomposition of A_0 [TODO: cite?]. A similar approach can be applied to the traditional Kronkecker product form (3), now transforming A_0 (instead of A_0^{-1}) using the real Schur decomposition. Solving for all stage vectors can then be achieved by solving a similar block upper triangular matrix as in (22), but now the 2×2 block linear systems corresponding to complex eigenvalues of A_0, say $\mu + i\zeta$, take the form

$$\begin{bmatrix} I - \mu\widehat{\mathcal{L}} & \zeta\widehat{\mathcal{L}} \\ -\zeta\widehat{\mathcal{L}} & I - \mu\widehat{\mathcal{L}} \end{bmatrix}.$$

The inversion of a 2×2 block matrix, via direct methods or preconditioning, is essentially defined by inverting the Schur complement, in this case given by

$$S_0 = (I - \mu\widehat{\mathcal{L}}) + \zeta^2(I - \mu\widehat{\mathcal{L}})^{-1}\widehat{\mathcal{L}}^2.$$

Interestingly, an analogous approach as used with the A_0^{-1} transformation can be applied here. Suppose we precondition $S_0 = (I-\mu\widehat{\mathcal{L}})+\zeta^2(I-\mu\widehat{\mathcal{L}})^{-1}\widehat{\mathcal{L}}^2$ with the term associated with the real-valued eigenvalue, $(I - \mu\widehat{\mathcal{L}})$. The preconditioned operator takes the form

$$\begin{aligned}(I - \mu\widehat{\mathcal{L}})^{-1}S_0 &= I + \zeta^2(I - \mu\widehat{\mathcal{L}})^{-2}\widehat{\mathcal{L}}^2 \\ &= I + \frac{\zeta^2}{\mu^2}(\mu I - \widehat{\mathcal{L}}^{-1})^{-2}.\end{aligned}$$

Note that this closely resembles the preconditioned operator from (9) and (25), but here with a $\widehat{\mathcal{L}}^{-1}$ inside of the inverse term, as opposed to a $\widehat{\mathcal{L}}$ in (9) and (25). In Theorem 3, we assumed that $\widehat{\mathcal{L}}$ is negative in a field of values sense, equivalent to saying that $(\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^T)$ is negative semi-definite. Noting that if $(\widehat{\mathcal{L}} + \widehat{\mathcal{L}}^T)$ is negative semi-definite, then $(\widehat{\mathcal{L}}^{-1} + \widehat{\mathcal{L}}^{-T})$ is also negative semi-definite, we have the following corollary.

{cor:fov}

COROLLARY 8 (Preconditioned field of values). *Assume that $\mu > 0$ and the symmetric part of $\widehat{\mathcal{L}}^{-1}$ satisfies $(\widehat{\mathcal{L}}^{-1} + \widehat{\mathcal{L}}^{-T}) \le 0$. Let $\mathcal{P}_\mu$ denote the preconditioned operator, where $(I - \mu\widehat{\mathcal{L}}) + \zeta^2(I - \mu\widehat{\mathcal{L}})^{-1}\widehat{\mathcal{L}}^2$ is preconditioned with $(\mu I - \widehat{\mathcal{L}})^{-1}$. Then $W(\mathcal{P}_\mu)$ is bounded by Ω as shown in Figure 1.*

Proof. The proof follows analogous to that of Theorem 3. ∎

Corollary 8 suggests the A_0^{-1} transformation is not necessary – one can apply the real Schur decomposition to the traditional Kronecker product form (3) and use a similar block preconditioning as introduced in (24) and (25) to solve the 2×2 blocks associated with complex eigenvalues, with similarly nice theoretical bounds on the conditioning of the preconditioned operator. Here, we use the A_0^{-1} approach as it *is* necessary for the linear algorithm developed in Section 2, and also more naturally suggests an extension to the general nonlinear setting (for example, full Newton as opposed to simplified Newton) in the following section.

onlinear:gen}

6.2. The general setting. Now let us return to (4) for $\mathcal{L}_i \neq \mathcal{L}_j$, but extract the real Schur decomposition as in Subsection 6.2. Continuing with the simplified representation $\widehat{\mathcal{L}}_i := \delta t M^{-1} \mathcal{L}_i$, this yields the linear system

$$\left(R_0 \otimes I - (Q_0^T \otimes I) \begin{bmatrix} \widehat{\mathcal{L}}_1 & & \\ & \ddots & \\ & & \widehat{\mathcal{L}}_s \end{bmatrix} (Q_0 \otimes I) \right) (R_0^{-1} Q_0^T \otimes I)\mathbf{k} = (Q_0^T \otimes I)\mathbf{f}. \tag{26}$$

Here, we are particularly interested in the operator

$$(Q_0^T \otimes I) \begin{bmatrix} \widehat{\mathcal{L}}_1 & & \\ & \ddots & \\ & & \widehat{\mathcal{L}}_s \end{bmatrix} (Q_0 \otimes I). \tag{27}$$

Of course for $\widehat{\mathcal{L}}_i = \widehat{\mathcal{L}}_j$, this is block diagonal, given by $I \otimes \widehat{\mathcal{L}}$. For $\widehat{\mathcal{L}}_i \neq \widehat{\mathcal{L}}_j$, diagonal blocks of (27) take the form $\sum_{k=1}^s |c_k| \widehat{\mathcal{L}}_k$, where $\sum_{k=1}^s |c_k| = 1$, and off-diagonal blocks take the form $\sum_{k=1}^s d_k \widehat{\mathcal{L}}_k$, where $\sum_{k=1}^s d_k = 0$.

Few thoughts:

- There are a number of ways we can think about this. We can pick some specific block triangular representation of (26) and invert it exactly for each nonlinear iteration. In the context of Newton, this would be a quasi-Newton type method. We could also use a block triangular approximation to (26) as a preconditioner and iterate on the linear system to invert the actual Jacobian. I think the preconditioning would be sufficiently good that this would converge very rapidly. The downside is that we would either need (i) to store preconditioners for each stage to be applied every outer linear iteration on the block system, or (ii) rebuild solvers for each stage during each linear iteration if we cannot store them all. Unclear what the "best" approach would be.
- An important piece of the puzzle is (27). A closed form as a function of $\{\widehat{\mathcal{L}}_k\}$ is straightforward to compute in Mathematica. In general, it seems for IRK schemes that diagonal blocks are dominated by one term, and effectively defined by two. Many of the off-diagonal elements have very small coefficients, and can probably be ignored without significant issue. But there are still many options. For example, we can include the upper triangular off-diagonal components of (27) at the expense of some extra mat-vecs. Is this additional work worth the improvement in convergence?
- Inverting the 2×2 diagonal blocks likely require the most careful attention. The off-diagonal blocks for a principle 2×2 submatrix in (27) are identical,

but the diagonal blocks differ. Do we take a linear combination $\sum_{k=1}^{s} |c_k| \widehat{\mathcal{L}}_k$ for some or all of the terms, or just use the dominant term? If we drop any terms, we should probably rescale to have coefficients sum to one. If we assume the off-diagonal blocks are zero, Corollary 8 extends well to a 2×2 matrix with different $\sum_{k=1}^{s} |c_k| \widehat{\mathcal{L}}_k$ on the diagonal blocks. If we have off-diagonal blocks, these will not commute, and the analysis may be more complicated.

References.

[1] R. Alexander, *Diagonally implicit runge–kutta methods for stiff ode's*, SIAM Journal on Numerical Analysis, 14 (1977), pp. 1006–1021.

[2] T. A. Bickart, *An Efficient Solution Process for Implicit Runge–Kutta Methods*, SIAM Journal on Numerical Analysis, 14 (1977), pp. 1022–1027, doi:10.1137/0714069.

[3] W. C. Brown, *Matrices over commutative rings*, Marcel Dekker, Inc., 1993.

[4] J. Butcher and D. Chen, *A new type of singly-implicit Runge–Kutta method*, Applied Numerical Mathematics, 34 (2000), pp. 179–188, doi:10.1016/s0168-9274(99)00126-9.

[5] J. C. Butcher, *On the implementation of implicit Runge-Kutta methods*, BIT Numerical Mathematics, 16 (1976), pp. 237–240, doi:10.1007/bf01932265.

[6] J. C. Butcher, *Numerical methods for ordinary differential equations*, John Wiley & Sons, 2016.

[7] K. E. Gustafson and D. K. Rao, *Numerical range*, in Numerical Range, Springer, 1997, pp. 1–26.

[8] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II, Stiff and Differential-Algebraic Problems*, (1996), pp. 118–130, doi:10.1007/978-3-642-05221-7_8.

[9] J. Liesen and P. Tichý, *The field of values bound on ideal gmres*, arXiv preprint arXiv:1211.5969, (2012).

[10] S. P. Nørsett, *Runge-kutta methods with a multiple real eigenvalue only*, BIT Numerical Mathematics, 16 (1976), pp. 388–393.

[11] B. Orel, *Real pole approximations to the exponential function*, BIT, 31 (1991), pp. 144–159, doi:10.1007/bf01952790.

[12] W. Pazner and P.-O. Persson, *Stage-parallel fully implicit Runge–Kutta solvers for discontinuous Galerkin fluid simulations*, Journal of Computational Physics, 335 (2017), pp. 700–717, doi:10.1016/j.jcp.2017.01.050.