IRK![*]

BEN S. SOUTHWORTH [†]

Abstract.

1. Introduction.

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

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

where M is a mass matrix, and $\mathcal{N} \in \mathbb{R}^{N\times N}$ 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 nodes $\mathbf{c}_0 = (c_0, \ldots, c_s)$.

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

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

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,

$$\left(\begin{bmatrix} M & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & M \end{bmatrix} - \delta t \begin{bmatrix} a_{11}\mathcal{L}_1 & \ldots & a_{1s}\mathcal{L}_1 \\ \vdots & \ddots & \vdots \\ a_{s1}\mathcal{L}_s & \ldots & a_{ss}\mathcal{L}_s \end{bmatrix} \right) \begin{bmatrix} \mathbf{k}_1 \\ \vdots \\ \mathbf{k}_s \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_s \end{bmatrix}. \tag{2}$$
{{eq:k0}}

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 to solve (2) are not a viable option, and fast, parallel iterative methods must be used.

[*] This research was conducted ...

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

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).

Nevertheless, there are a number of desirable properties of IRK schemes, particularly in terms of accuracy and stabillity. Practical alternatives to IRK schemes include diagonally implicit Runge-Kutta methods (DIRK), where A_0 is lower triangular, or singly implicit Runge Kutta methods (SIRK), where A_0 has exactly one positive real eigenvalue. For such schemes, the solution of (2) only requires s linear solves 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 nonlinear PDEs, it is typically the case that the global order of accuracy is limited to $\approx \min\{p, q+1\}$, for integration order p and stage-order q. In addition [TODO: ref stability]. Thus, actually getting high-order accuracy with stiff PDEs more-or-less requires the use of IRK methods.

A common approach 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. For nonlinear problems, this corresponds to a simplified Newton method, where the Jacobian is only evaluated at one time-point per time step (or also naturally arises for a linear problem with no time-dependence in spatial differential components). This 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 real-valued spatial operator or Jacobian independent of time. Here, we revive some of the older Runge-Kutta work in the light of a changing computational landscape, developing a new iterative approach to solve (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 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.

ALso maybe useful in solving SDIRK because $\eta \gg a_{ij}$??

[TODO: outline sections]

c:intro:prev}

1.2. Why IRK and previous work. The Kronecker product form (resulting from the assumption that $\mathcal{L}$ is independent of time) allows for further simplifications. First proposed in [TODO: Butcher77], let A_0 have 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 and lower triangular entries corresponding to the (unit-valued) Jordan blocks. Then (3) is equivalent to the problem

$$(U_0 \otimes I)(I \otimes M - \delta t L_0 \otimes \mathcal{L})(U_0^{-1} \otimes I)\mathbf{k} = \mathbf{f}, \tag{4}$$

$$(I \otimes M - \delta t L_0 \otimes \mathcal{L})\hat{\mathbf{k}} = \hat{\mathbf{f}}, \tag{5}$$

where $\hat{\mathbf{f}} := (U_0^{-1} \otimes I)\mathbf{f}$ and $\hat{\mathbf{k}} := (U_0^{-1} \otimes I)\mathbf{k}$. Note that computing the Jordan decomposition of A_0 is typically of trivial computational expense compared to other operations. The transformed system in (4) can now be inverted by applying a block diagonal or lower triangular solve, which only requires inverting the diagonal blocks, $M - \delta_t(L_0)_{ii}\mathcal{L}$.

Transforming (3) into (4) reduces the solution of an $ns \times ns$ system of equations to solving s linear systems of size $n \times n$. The downside is that the IRK schemes with high accuracy and stability have primarily complex eigenvalues, in which case the original real-valued system was transformed into a set of smaller complex systems (if the original matrix $\mathcal{L}$ is complex, obviously complex eigenvalues are not a concern). Complex eigenvalues introduce several difficulties. First, complex operations require several times the floating-point operations to perform standard algebraic operations. Second, not all preconditioners are well-developed for complex-valued matrices. Last, an important practical concern is that not all software libraries support complex numbers.

Published shortly after (and independently from) Butcher's result regarding Jordan normal form [TODO: cite], Bickart developed a similar result [TODO: cite]. If we define $Q_s(x)$ as the characteristic polynomial of A_0, then the inverse of (3) can be computed via some matrix-vector multiplications and the action of $Q_s(\mathcal{L})^{-1}$ [TODO: cite]. This is similar in principle to Butcher's result, as in practice one could 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 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.

[TODO: Fix discussion on Bickart's result, work out what he actually did carefully.]

Butcher (76), Bickart (77)

- Uses tensor product structure, gets Jordan normal form of Butcher matrix A. Reformulates problem as block subdiagonal, with diagonal blocks $I - h\lambda J$, for Jacobian J, time step h, and eigenvalue λ of A (are we sure not A^{-1}??).
- Must solve for complex eigenvalues. This is not desirable.
- Bickart uses polynomial form, closer to what I look at. Originally based on Tensor product result from Jameson (68)

Varah (79):

- Much better description of Butcher's method. Transforms coordinate system of Jacobian tensor. Still relies on this tensor structure to do said transformation.
- Transforms Jacobian to Hessenberg form to avoid repeated action computing LU of a shifted Jacobian.

Burrage (82):

- Look at stability of DIRK and SIRK methods, which have one eigenvalue and are more amenable to development by Butcher.

Orel (91):

- Looks at RK methods w/ real eigenvalues. Turns out best approximation to exponential is obtained by having all eigenvalues equal (SIRK/SDIRK methods). SIRK methods offer some advantages over SDIRK methods, but lack the favorable stabillity an accuracy of IRK methods.

Cooper:

- (83,90,93): develops single Newton scheme using SOR or various matrix splittings, applied to ODEs. This is a single-step Newton method, where the actual Jacobian solve is replaced with an SOR iteration. Here, Butcher matrix A is replaced with an approximate matrix with one positive real eigenvalue.

Pinto:

- (95) Mostly ODEs, do consider 1d-space-1d-time burgers on a very small grid.
- (96) additional iteration of single Newton, makes it quasi-Newton like.
- (01) Analysis of single Newton (SOR) w/ simplified Newton for higer order.

Butcher (2000):

- Develops ESDIRK schemes w/ higher stage order and stiff accuracy than traditional SIRK/DIRK schemes. Based on using iniital explicit stage, moves RK abscissae.

Brugano (2014) and Antonana (2018)

- New splitting and IRK techniques for Hamiltonian problems where conservation is important. Used for ODEs.
- One obvious drawback – even if spatial operator/Jacobian is SPD, IRK system is nonsymmetric.

Lay (2000)

-

Van Lent (2004)

- Multigrid for IRK?

Staff & Mardal (2006)

- One of first paper to consider preconditioning the fully implicit RK system. Use block Jacobi and block lower triangular preconditioners for the diffusion equation. Use multigrid V-cycles and full Newton time-dependent Jacobian. Upper triangular is bad compared to block Jacobi and lower triangular (this has appeared elsewhere in literature – A_0 is dominant in lower triangular part – Crout factorization in I think Messina or Van der Houwen; comes up again in Brugano (2015)).

Mardal (2007)

- Analyze block-diagonal preconditioners in a Sobolev setting, demonstrate conditioning of the preconditioned operator to be optimal in the independent of h sense. Use multigrid w/ diffusion as example.

Nilssen (2011)

- Analogous to above, Sobolev analysis for block-diagonal preconditioning applied to the bidomain equations.

Xie (2011)

- Proposes a modified simplified Newton for the time-dependent cast, where the Jacobian is formed based on a least squares approximation to the true RK coefficients, evaluating all entries at a single time point. In example problems, modified Jacobian typically converged faster than simplified (evaluated at previous time step), up to 2x less iterations/time.

Hao Chen:

- (2014) Develops a splitting iterative method to precondition IRK matrices, similar to ADI schemes. Proves that for definite spatial operators and Butcher matrices (that is, eignvalues have positive or negative real parts), $\rho(T) < 1$, where T is the fixed-point iteration matrix. Look at diffusion equation with IRK and BVMs.
- (2016) Analogous to above, extended to wave equation.

Pazner

-

2. Fast parallel solvers for IRK.

The RK stage system in (2) can be reformulated as [TODO: cite Will]

$$\left(A_0^{-1}\otimes M - \delta t\begin{bmatrix}\mathcal{L}_1 & & \\ & \ddots & \\ & & \mathcal{L}_s\end{bmatrix}\right)(A_0\otimes I)\begin{bmatrix}\mathbf{k}_1\\ \vdots\\ \mathbf{k}_s\end{bmatrix} = \begin{bmatrix}\mathbf{f}_1\\ \vdots\\ \mathbf{f}_s\end{bmatrix}. \tag{6}$$

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 $\mathcal{L}_i \mapsto \delta t M^{-1}\mathcal{L}_i$, $i = 1,..,s$. Note the time step δt for the given Runge-Kutta step is now included in $\mathcal{L}_i$. 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

{{eq:k1}}

$$\begin{aligned}\mathcal{M}_s &:= A_0^{-1}\otimes I - \begin{bmatrix}\mathcal{L}_1 & & \\ & \ddots & \\ & & \mathcal{L}_s\end{bmatrix}\\ &= \begin{bmatrix}\alpha_{11}I-\mathcal{L}_1 & \alpha_{12}I & ... & \alpha_{1s}I\\ \alpha_{21}I & \alpha_{22}I-\mathcal{L}_2 & & \alpha_{2s}I\\ \ddots & & \ddots & \vdots\\ \alpha_{s1}I & ... & \alpha_{s(s-1)}I & \alpha_{ss}I-\mathcal{L}_s\end{bmatrix}.\end{aligned} \tag{7}$$

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 (7) but based on a principle submatrix of A_0.

{sec:solve:inv}

2.1. An inverse and update for commuting operators.

This section introduces a result similar to Bickart's but generalized to hold for commuting operators. If $\mathcal{L}_i = \mathcal{L}_j$ for all i,j, we show that the inverse of (7) can be expressed in terms of $P_s(\mathcal{L})^{-1}$, where $P_s(\mathcal{L})$ is the characteristic polynomial of A_0^{-1}.

{lem:inv}

Lemma 1. *Let α_{ij} denote the (i,j)th entry of A_0^{-1} and assume $\{\mathcal{L}_i\}_{i=1}^s$ are commuting operators. Define $\mathcal{M}_s$ as in (3). 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 $\mathcal{D}_s$ is invertible, and*

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

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

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

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

Proof. Notice in (7) that if $\mathcal{L}_i$ and $\mathcal{L}_j$ commute for all i,j, then $\mathcal{M}_s$ is a matrix over the commutative ring of linear combinations of I and $\{\mathcal{L}_i\}$. 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 $\mathcal{M}_s^{-1} := \det(\mathcal{M}_s)^{-1}\mathrm{adj}(\mathcal{M}_s)$. [TODO: Need citations!] For the case of time-independent operators ($\mathcal{L}_i = \mathcal{L}_j$), notice that $\mathcal{M}_s$ takes the form $A_0^{-1} - \mathcal{L}I$ over the commutative ring defined above. Analogous to a scalar matrix, the determinant of $A_0^{-1} - \mathcal{L}I$ is the characteristic polynomial of A_0^{-1} evaluated at $\mathcal{L}$. □

Returning to (6), we can express the direct solution for the set of all stage vectors $\mathbf{k} = [\mathbf{k}_1; ...; \mathbf{k}_s]$ as

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

where $\mathbf{f} = [\mathbf{f}_1; ...; \mathbf{f}_s]$ (note that $A_0 \otimes I$ commutes with $\det(M_s)^{-1}$). Excusing the slight abuse in notation, let $\det(M_s)^{-1}$ now denote just the diagonal block (rather than a block-diagonal matrix). The Runge-Kutta update is then given by

$$\begin{aligned} \mathbf{u}_{n+1} &= \mathbf{u}_n + \delta t \sum_{i=1}^{s} b_i \mathbf{k}_i \\ &= \mathbf{u}_n + \delta t \det(M_s)^{-1}(\mathbf{b}_0^T A_0^{-1} \otimes I)\mathrm{adj}(\mathcal{M}_s)\mathbf{f}. \end{aligned} \tag{8}$$

The adjugate simply consists of linear combinations of I and $\mathcal{L}$, 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 $\mathcal{L}$ of degree $s - 1$ (or linear combinations of comparable degree if $\mathcal{L}_i \neq \mathcal{L}_j$) and off-diagonal terms are polynomials in $\mathcal{L}$ of degree $s - 2$.

Returning to (8), we consider two cases. First, if a given Runge-Kutta scheme is stiffly accurate, then $\mathbf{b}_0^T A_0^{-1} = [0, ..., 0, 1]$. This yields the nice simplification that computing the update in (8) only requires applying the last row of $\mathrm{adj}(\mathcal{M}_s)$ to $\mathbf{f}$ (in a dot product sense) and applying $\det(M_s)^{-1}$ 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 (e.g., Gauss methods), one can obtain (again using, e.g., Mathematica) 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 $\mathcal{L}$ 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.

2.2. Stability and the method of lines.

- A necessary condition for stable time-propagation is that eigenvalues of $\mathcal{L}$ lie in the region of stability for the given RK scheme. For virtually all implicit RK schemes, this means that eigenvalues of $\mathcal{L}$ 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 $\mathcal{L}$ 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 $\mathcal{L}$ has a non-positive field of values for stability?

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

In contrast to some of the original work on solving IRK systems, where LU factorizations were the dominant cost and system sizes relatively small [TODO: cite], explicitly forming and inverting $P_s(\mathcal{L})$ 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 $P_s(\mathcal{L})$ in a factored form,

(9) {{eq:fac}} $$P_s(\mathcal{L}) = \prod_{i=1}^{s} (\lambda_i I - \mathcal{L}),$$

and its inverse can then be computed by successive applications of $(\lambda_i I - \mathcal{L})^{-1}$, 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 $(\lambda_i I - \mathcal{L})^{-1}$ more difficult and often not practical.

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

(10) {{eq:imag1}} $$((\eta + i\beta)I - \mathcal{L})((\eta - i\beta)I - \mathcal{L}) = (\eta^2 + \beta^2)I - 2\eta\mathcal{L} + \mathcal{L}^2.$$

In practice, we typically do not want to directly form or precondition a quadratic operator like (10). For example, a discretization of diffusion is typically conditioned like $1/h^2$ for mesh spacing h (worse for high-order discretizations), so considering $\mathcal{L}^2$ directly results in conditioning $\sim \mathcal{O}(1/h^4)$. Moreover, many fast parallel methods such as multigrid are not well-suited for solving a polynomial in $\mathcal{L}$. The point of (10) is that by considering conjugate pairs of eigenvalues, the resulting operator is real-valued. Thus, consider preconditioning (10) with the inverse of the real-valued quadratic polynomial, $(\eta I - \mathcal{L})(\eta I - \mathcal{L})$. Expanding, the preconditioned operator takes the form

$$\begin{aligned} \mathcal{P}_A &:= (\eta I - \mathcal{L})^{-2} \left[(\eta^2 + \beta^2)I - 2\eta\mathcal{L} + \mathcal{L}^2\right] \\ &= I + \beta^2(\eta I - \mathcal{L})^{-2} = I + \frac{\beta^2}{\eta^2}\left(I - \tfrac{1}{\eta}\mathcal{L}\right)^{-2}. \end{aligned}$$ (11) {{eq:prec1}}

It turns out, by assumptions on $\mathcal{L}$ that yield a stable time-propagation scheme, the preconditioned operator in (11) is very well-conditioned. Theorem 2 analyzes the field-of-values of $\mathcal{P}_A$, denoted $W(\mathcal{P}_A)$, as a measure of the preconditioning, and Corollary 3 extends this preconditioning to prove fast convergence of GMRES.

{th:fov}

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

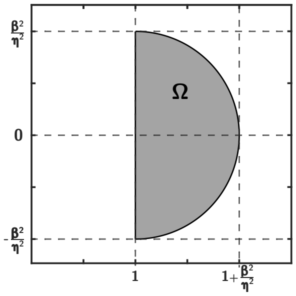

Fig. 1. {fig:bound}

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

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

Note that an exact inverse yields $\mathcal{P}_A = I$, with spectrum and field-of-values given by $\sigma(\mathcal{P}_S) = W(\mathcal{P}_S) = \{1\}$. Appealing to (11) and the final property stated above, $W(\mathcal{P}_A) = 1 + \frac{\beta^2}{\eta^2} W(E)$, for error term $E := (I - \frac{1}{\eta}\mathcal{L})^{-2}$ 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 $\mathcal{L}$ satisfies $(\mathcal{L} + \mathcal{L}^T)/2 \leq 0$. This is somehow related/necessary to RK stability?? It follows that the real part of eigenvalues of $\mathcal{L}$ are non-positive and, thus, $(I - \frac{1}{\eta}\mathcal{L})$ cannot have a zero eigenvalue and must be invertible. Furthermore, it also follows that the symmetric part of $(I - \frac{1}{\eta}\mathcal{L})$ is symmetric positive definite and thus the symmetric part of $(I - \frac{1}{\eta}\mathcal{L})^{-2}$ 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 $(\mathcal{L} + \mathcal{L}^T)/2 \leq 0$, we have

$$\frac{\left\langle (I - \frac{1}{\eta}\mathcal{L})\mathbf{x}, (I - \frac{1}{\eta}\mathcal{L})\mathbf{x} \right\rangle}{\langle \mathbf{x}, \mathbf{x} \rangle} = 1 - \frac{\langle (\mathcal{L} + \mathcal{L}^T)\mathbf{x}, \mathbf{x} \rangle}{\eta \langle \mathbf{x}, \mathbf{x} \rangle} + \frac{\langle \frac{1}{\eta^2}\mathcal{L}^T\mathcal{L}\mathbf{x}, \mathbf{x} \rangle}{\langle \mathbf{x}, \mathbf{x} \rangle} \geq 1 \tag{12}$$

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

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

This yields a bound on the numerical radius $r(E) = r((I - \frac{1}{\eta}\mathcal{L})^{-2}) \leq \|(I - \frac{1}{\eta}\mathcal{L})^{-2}\| \leq 1$. 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 3. *Let π_k denote the set of consistent polynomials of degree k. Then the ideal GMRES bound (an upper bound in operator norm on worst-case convergence) on convergence after k iterations applied to the preconditioned operator $\mathcal{P}_A$ (11) is bounded by*

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

Proof. For operator M, let $\nu(M)$ denote the distance of $W(M)$ from the origin. In [TODO: cite Liesen], they define $\cos(\beta) := \nu(M)/r(M)$, and prove that worst-case convergence of GMRES applied to operator M is bounded by (see Lemma 3.2)

$$\min_{p\in\pi_k} \|p(M)\| \leq 2\left(\frac{1-\cos\beta}{1+\cos\beta}\right)^k. \tag{13}$$

{eq:gmres}

For $M = \mathcal{P}_A$, we have $\nu(\mathcal{P}_A) = 1$ and $r(\mathcal{P}_A) \leq 1+\beta^2/\eta^2$. Plugging into (13) completes the proof. □

- Comment on size of η and β and convergence bounds for GMRES. Largest (squared) ratio I have seen is ~ 4.29 for RadauIIA(11), but another eigenvalue pair for the same scheme had ratio ~ 0.04, so the seem to even out.
- Is $\eta > 0$ always? Looks like it for everything I have tested, would be nice to say something general.. If not, need to confirm it holds for lots of standard methods. This is important assumption in above analysis.
- Classical GMRES convergence results based on $\lambda_{\min}((\mathcal{P}_A+\mathcal{P}_A^T)/2)$ and $\lambda_{\max}(\mathcal{P}_A^T\mathcal{P}_A)$ can also be applied, yielding a less tight result of $\left(\frac{\beta^2/\eta^2}{1+\beta^2/\eta^2}\right)^{k/2}$. The field-of-values is $2-4\times$ tighter, and deriving the field-of-values also reflects the nice conditioning of the operator.

2.4. Preconditioning quadratic polynomials. In practice, we generally do not want to directly form preconditioners for a quadratic polynomial in $\mathcal{L}$. Even forming such an operator can be expensive in parallel. Moreover, many well-known fast parallel preconditioners such as multigrid would likely struggle when applied directly to a matrix quadratic. Instead, the previous section suggests we can use two successive applications of a preconditioner for $(\eta I-\mathcal{L})$. If $(\eta I-\mathcal{L})^{-1}$ is applied twice to the *action* of $(\eta^2+\beta^2)I-2\eta\mathcal{L}+\mathcal{L}^2$, Theorem 2 proves that the preconditioned operator is very well conditioned and GMRES will converge rapidly (see also Corollary 3). However, in practice, fully converging $(\eta I-\mathcal{L})^{-1}$ is not desirable – even if GMRES converges rapidly, if each iteration requires a full linear solve, the resulting method remains moderately expensive. Here, we propose applying GMRES to $(\eta^2+\beta^2)I - 2\eta\mathcal{L}+\mathcal{L}^2$ by computing the operator's action (that is, not fully constructing it), and preconditioning each GMRES iterations with *two* applications of a sparse parallel preconditioner for $(\eta I-\mathcal{L})$, representing the action of $(\eta I-\mathcal{L})^{-2}$.

- I am thinking each GMRES iteration we only do a few (or even one?) AMG iterations for each application of $(\alpha I-\mathcal{L})$, hoping that at the end it doesn't really take more iterations than if we were to directly apply AMG preconditioned GMRES to $(\alpha I-\mathcal{L})^2$.
- We are actually preconditioning a much worse-conditioned operator. The $+\beta^2$ term we get in the operator (rather than preconditioner) shifts the field-

of-values positive and away from the origin by β. Hopefully this pans out by observing rapid convergence..

- One thing I am really puzzled by – by formulating based on A_0^{-1}, the eigenvalues in my tests appear to be fairly large in magnitude. E.g., eigenvalues of A_0 are ~ 0.1 and eigenvalues of A_0^{-1} are ~ 5. If this is true and we can get away with solving $(\eta I - \mathcal{L})$ for $\eta \sim 10\times$ larger than eigenvalues of A_0, that could be huge. In theory such systems would likely be way better conditioned. It also seems too good to be true/not very intuitive to solve a system with such a big shift when our actual time step is $\ll$..