Electronic Transactions on Numerical Analysis.
Volume 60, pp. 381–404, 2024.

ISSN 1068–9613.
DOI: 10.1553/etna_vol60s381

ETNA
Kent State University and
Johann Radon Institute (RICAM)

POLYNOMIAL PRECONDITIONING FOR THE ACTION OF THE MATRIX SQUARE ROOT AND INVERSE SQUARE ROOT*

ANDREAS FROMMER†, GUSTAVO RAMIREZ-HIDALGO†, MARCEL SCHWEITZER†, AND MANUEL TSOLAKIS†

Abstract. While preconditioning is a long-standing concept to accelerate iterative methods for linear systems, generalizations to matrix functions are still in their infancy. We go a further step in this direction, introducing polynomial preconditioning for Krylov subspace methods that approximate the action of the matrix square root and inverse square root on a vector. Preconditioning reduces the subspace size and therefore avoids the storage problem together with—for non-Hermitian matrices—the increased computational cost per iteration that arises in the unpreconditioned case. Polynomial preconditioning is an attractive alternative to current restarting or sketching approaches since it is simpler and computationally more efficient. We demonstrate this for several numerical examples.

Key words. polynomial preconditioning, matrix square root, inverse square root, Krylov space, matrix functions

AMS subject classifications. 65F60, 65F08, 65F50, 15A16

1. Introduction.

Even if a matrix $A \in \mathbb{C}^{n\times n}$ is sparse, its matrix function $f(A) \in \mathbb{C}^{n\times n}$, with $f : D \subseteq \mathbb{C} \to \mathbb{C}$ an appropriate function, is typically a full matrix. This is why for large A one has to resort to computing the action $f(A)b$ for some vector b rather than the full matrix $f(A)$—and in most applications, the action is typically all that is required.

Polynomial or rational Krylov subspace methods are then the methods of choice. Rational Krylov methods require the repeated solution of systems with coefficient matrices $A - \sigma I$ and various shifts σ. If no efficient solvers are available for such systems, then polynomial methods are the only viable approach, and it is these methods that we consider in the present paper.

Polynomial Krylov subspace methods use a nested orthonormal basis $v_1, \ldots, v_m$ of the Krylov subspaces $\mathcal{K}_m(A, b) = \text{span}\{b, Ab, \ldots, A^{m-1}b\}$, $m = 1, 2, \ldots$, from which they then typically extract the *Arnoldi approximation* f_m for $f(A)b$ as

$$f_m = V_m\, f(V_m^* A V_m)\, V_m^* b, \qquad \text{where } V_m = [v_1|\cdots|v_m] \in \mathbb{C}^{n\times m}.$$

Herein, $V_m^* A V_m =: H_m \in \mathbb{C}^{m\times m}$ is an upper Hessenberg matrix, the entries of which arise in the orthogonalization process when determining the basis vectors $v_1, \ldots, v_m$, and $V_m^* b = e_1 \|b\|$, with e_1 the first canonical unit vector in $\mathbb{C}^m$ and $\|\cdot\|$ the Euclidean vector norm.

In the non-Hermitian case, the orthogonalization costs of the Arnoldi process can become prohibitive if a large number of iterations is required. But even in the Hermitian case there is no short recurrence formula to update f_{m+1} from f_m (except for the linear system case, i.e., $f(z) = z^{-1}$), which means that all Arnoldi vectors must be stored.

For these reasons, approaches which aim at limiting the dimension m of the Krylov subspace by using *restarts* have attracted a lot of attention in the last two decades [1, 22, 23, 28, 29, 42, 64]; see also [35, 36] (and the references therein) for an overview of limited-memory methods for $f(A)b$ in general. The idea is to express the error $f(A)b - f_m$ as the action of a new matrix function $\widehat{f}(A)\widehat{b}$ on a new vector $\widehat{b}$ and recurse. Several representations for $\widehat{f}$ have been studied, and the challenge is to formulate one which is numerically stable. This is

*Received January 12, 2024. Accepted April 18, 2024. Published online on July 2, 2024. Recommended by Kathryn Lund.

†Department of Mathematics, Bergische Universität Wuppertal, 42097 Wuppertal, Germany ({frommer,ramirezhidalgo,marcel,tsolakis}@uni-wuppertal.de).

ETNA
Kent State University and
Johann Radon Institute (RICAM)

possible if f is a Stieltjes transform [2], a Cauchy-type function as defined in [29], or a Laplace transform [30]. In all these cases, $\widehat{f}$ is given as an integral, and sufficiently precise numerical quadrature has to be employed in order to evaluate $\widehat{f}$ and the corresponding functions occurring after more than one restart.

Another approach that has recently emerged for improving the performance of Krylov methods for functions of nonsymmetric matrices is to use a *truncated Arnoldi process*, thus mimicking the short recurrence available in the symmetric case. This yields local orthogonality of the basis vectors only, and the approach then combines this with *randomized sketching* [69], which allows one to subsequently perform a cheaper (implicit) orthogonalization of the Krylov basis, thus potentially reducing arithmetic operations, storage, and communication [7, 17, 37, 53]. These sketching methods can often be applied very successfully, in particular for entire functions like the exponential; see [55] for a theoretical justification. For other functions, sketched Krylov methods may fail completely though, as spurious Ritz values outside the field of values of A can cause the methods to essentially break down when they hit a singularity or branch cut of f; see [17, Section 5.1.3] and [37, Section 5.4] for examples of this phenomenon.[1]

For the functions $f(z) = z^{-1/2}$ and $f(z) = z^{1/2}$, we here propose polynomial preconditioning as an alternative to keep the Krylov dimension m—and thus the orthogonalization cost—small. We extract an approximation of $f(A)b$ from a Krylov subspace $\mathcal{K}_m(Ap_{k-1}(A), c)$, where p_{k-1} is an appropriate polynomial of degree $k-1$ and c is a suitably chosen starting vector. The fact that the square root has two branches imposes constraints on p_{k-1} and A, as we will discuss in detail in Section 5, covering the practically important case where the spectrum of A is contained in the right half-plane. While polynomial preconditioning as a technique for linear systems and eigenvalue computations has been used and investigated for a long time [4, 21, 27, 43, 45, 46, 60, 61, 65], we believe that its extension to the inverse square root and the square root are novel, and our numerical experiments will show that polynomial preconditioning can substantially outperform restarted and sketched approaches.

Since $\mathcal{K}_m(Ap_{k-1}(A), b) \subseteq \mathcal{K}_{mk}(A, b)$, polynomial preconditioning extracts its approximation from a smaller subspace than the unpreconditioned method when investing the same number of matrix-vector multiplications. In this sense, the polynomially preconditioned approximation can be expected to be less accurate than the unpreconditioned Arnoldi approximation for the same investment of matrix-vector multiplications. This is actually a theorem for the linear system case and for variational methods that minimize a measure for the error, like the CG or GMRES methods. The possible gains with polynomial preconditioning for matrix functions reside in the fact that (i) they allow to reduce storage (for Hermitian and non-Hermitian matrices), (ii) they reduce arithmetic cost due to orthogonalization (for non-Hermitian matrices), which can become prohibitive if many iterations are needed, (iii) they reduce communication cost on parallel machines where orthogonalization requires global communication, (iv) they avoid restarts which are non-trivial to implement for matrix functions, and (v) they can exhibit the typical superlinear convergence behavior of Krylov subspace methods, which is usually lost with restarts.

The square root and inverse square root of a matrix have many applications in a variety of scientific computing and engineering applications. For example, the square root and inverse square root of discretized differential operators arise when computing Dirichlet-to-Neumann and Neumann-to-Dirichlet maps [3, 20], while fractional powers of the graph Laplacian are used for modeling anomalous diffusion and other non-local phenomena in complex networks [10, 11, 25]. Applications in data science include sampling from Gaussian Markov

[1] We briefly mention that another "flavor" of sketched Krylov methods exists which reduces orthogonalization cost, too, by employing a sketched inner product, possibly together with mixed precision [5, 6, 17, 66]. Such methods, however, cannot overcome the quadratic dependence of the cost on the iteration number.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Random fields [56] and whitening to increase the fidelity of stochastic variational Gaussian processes [57], which require inverse square roots of precision and kernel matrices. Due to the relation $\operatorname{sign}(z) = z(z^2)^{-1/2}$, the inverse square root is also often used in applications where the action of the matrix sign function is needed, e.g., when working with the overlap Dirac operator in lattice quantum chromodynamics [15, 67].

To conclude this introduction, let us mention that in the literature two other types of techniques have been proposed which can be regarded as preconditioning for matrix functions in a broader sense. The shift-invert (or RD-rational) Lanczos method [51, 68] is a special case of more general rational Krylov subspace methods. It thus requires the solution of a shifted linear system in each iteration, a situation that we consider infeasible in the context of this paper. Additionally, [57] proposes a preconditioning technique for the (inverse) matrix square root which alters the resulting vector: The method does not return $f(A)b$, but only a vector which agrees with $f(A)b$ up to certain rotations. While such a vector is all that is needed for the specific application considered in [57], it is not possible to use this kind of approach when the vector $f(A)b$ itself is required (as is typically the case).

This paper is organized as follows: In Section 2 we shortly review left and right preconditioning for linear systems and then show how this can be extended to multiplicative matrix functions and polynomial preconditioners. In Section 3 we provide algorithmic details of preconditioned methods for the inverse square root and give a theoretical justification for why they can be expected to greatly improve convergence speed. We discuss the extension to the square root in Section 4. Sensible choices for preconditioning polynomials and how to evaluate them at a matrix argument are considered in Section 5. We then report results for several numerical experiments in Section 6 before ending with our conclusions in Section 7.

2. Polynomial preconditioning. Preconditioning is a well-established technique for solving linear systems of equations, i.e., when $f(z) = z^{-1}$. For any nonsingular matrix M, we have that

$$A^{-1}b = (M^{-1}A)^{-1}M^{-1}b = M^{-1}(AM^{-1})^{-1}b. \tag{2.1}$$

The first equality in (2.1) gives rise to left preconditioning, where we compute approximations x_m to $A^{-1}b$ from the Krylov subspaces $\mathcal{K}_m(M^{-1}A, M^{-1}b)$, and the second equality to right preconditioning, where we obtain approximations $x_m = M^{-1}y_m$ with y_m from the Krylov subspace $\mathcal{K}_m(AM^{-1}, b)$. The challenge is to find a preconditioner such that $M^{-1}u$ is relatively easy to compute for any vector u and at the same time the preconditioned matrix AM^{-1} or $M^{-1}A$ is close enough to the identity such that a typical Krylov subspace method will take far fewer iterations to converge than the same method using just the matrix A.

The inverse $f(z) = z^{-1}$ has the property that $(z_1z_2)^{-1} = z_1^{-1}z_2^{-1} = z_2^{-1}z_1^{-1}$, and as a matrix function this translates into the very peculiar property that for any two nonsingular matrices A and B we have

$$(AB)^{-1} = B^{-1}A^{-1}, \qquad (BA)^{-1} = A^{-1}B^{-1},$$

which is at the origin of the equalities (2.1) used for preconditioning. The order of the factors matters unless A and B commute, in which case $f(A)g(B) = g(B)f(A)$ for any two functions f and g and thus in particular for $f(z) = g(z) = z^{-1}$; cf. [40].

If we want to transfer the idea of preconditioning to functions f other than z^{-1}, one path to follow is to identify situations in which $f(AB)$ can easily be connected to $f(A)$ and/or $f(B)$. The following proposition gives such a result for f a (possibly non-integer) power of a matrix A and B a polynomial in A.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

PROPOSITION 2.1. *Let $A \in \mathbb{C}^{n\times n}$, let p be a polynomial, and consider the function z^α for some $\alpha \in \mathbb{R}$. If $\alpha < 0$, then further assume that the matrices A and $p(A)$ do not have eigenvalues in $(-\infty, 0]$. Then,*

$$(Ap(A))^\alpha = A^\alpha (p(A))^\alpha = (p(A))^\alpha A^\alpha, \tag{2.2}$$

Proof. By [40, Theorem 1.17], we have that if $f(z) = g(h(z))$, then $f(A) = g(h(A))$, assuming that g and h are such that $h(A)$ and $g(h(A))$ are well defined. We apply this result with $g(z) := z^\alpha$ and $h(z) := zp(z)$, writing $f(z) = g(h(z)) = (zp(z))^\alpha = z^\alpha (p(z))^\alpha$, which immediately gives the first equality in (2.2). The second equality follows because A^α and $(p(A))^\alpha$ commute; see [40, Theorem 1.13(a), (e)]. ∎

For the functions considered in Proposition 2.1 we thus have, assuming that $p(A)$ and thus $(p(A))^\alpha$ is nonsingular, that

$$A^\alpha b = (Ap(A))^\alpha (p(A))^{-\alpha} b = (p(A))^{-\alpha} (Ap(A))^\alpha b.$$

From this we can get, at least in principle, a left polynomially preconditioned method for $A^\alpha b$ by extracting the iterates f_m as the Arnoldi approximation from the Krylov subspace $\mathcal{K}_m(Ap(A), p(A)^{-\alpha} b)$ and a right polynomially preconditioned method by taking $f_m = (p(A))^{-\alpha} \widetilde{f}_m$ with $\widetilde{f}_m$ being the Arnoldi approximations from the Krylov subspace $\mathcal{K}_m(Ap(A), b)$. However, this becomes a computationally feasible approach only if the action of $(p(A))^{-\alpha}$ on a vector is easy to compute. This is possible for special choices of p and $\alpha = -1/2$, the inverse square root, and can be extended to the square root $\alpha = 1/2$ as we will describe in the next sections.

3. Polynomial preconditioning for the inverse square root. In this section, we first describe algorithms for left and right polynomial preconditioning of the inverse square root. We then analyze the effect that preconditioning has on the spectrum (and thus the condition number) of the Hermitian positive definite A.

In a polynomially preconditioned method for the inverse square root, we want $p(A)$ to approximate A^{-1} so that $Ap(A)$ is close to the identity. At the same time, $(p(A))^{1/2}$ needs to be "easy" to evaluate. We thus take $p(z) = (q(z))^2$, where q is chosen as a polynomial that approximates $z^{-1/2}$ to achieve both goals at the same time. The relation (2.2) becomes, for $\alpha = -1/2$,

$$A^{-1/2} b = (A(q(A))^2)^{-1/2} q(A) b = q(A)(A(q(A))^2)^{-1/2} b,$$

provided we have

$$\left((q(A))^2\right)^{1/2} = q(A). \tag{3.1}$$

Whether (3.1) holds or not depends on the branch that we take for the square root and on the distribution of the eigenvalues of A. We will always assume that we take the principal branch of the square root given as

$$z = |z| e^{i \arg(z)} \mapsto \left|\,|z|^{1/2}\right| e^{i \arg(z)/2}, \qquad \text{for } \arg(z) \in (-\pi, \pi],$$

where $i = \sqrt{-1}$, i.e., we put the branch cut on the negative real axis. Then, for any polynomial q we have $\left((q(z))^2\right)^{1/2} = q(z)$ if and only if $\arg(q(z)) \in (-\pi/2, \pi/2]$, and thus

$$\left((q(A))^2\right)^{1/2} = q(A) \qquad \text{if } \operatorname{spec}(q(A)) \subseteq \mathbb{C}^+, \tag{3.2}$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Algorithm 1 m steps of left polynomially preconditioned Arnoldi for $A^{-1/2}b$.

1: choose polynomial q such that $q(A)$ approximates $A^{-1/2}$
2: put $c \leftarrow q(A)b$, $v_1 \leftarrow c/\|c\|$
3: **for** $j = 1, \ldots, m$ **do** ▷ Arnoldi process for preconditioned matrix
4: compute $u \leftarrow Av_j$, $y \leftarrow q(A)u$, $w \leftarrow q(A)y$
5: **for** $i = 1, \ldots, j$ **do**
6: $h_{ij} \leftarrow \langle w, v_i \rangle$, $w \leftarrow w - v_i h_{ij}$ ▷ orthogonalize against previous vectors
7: **end for**
8: $h_{j+1,j} \leftarrow \|w\|$
9: $v_{j+1} \leftarrow w/h_{j+1,j}$
10: **end for**
11: $f_m \leftarrow V_m(H_m^{-1/2}e_1\|c\|)$ ▷ $V = [v_1|\cdots|v_m]$, $H_m = (h_{ij}) \in \mathbb{C}^{m\times m}$ upper Hessenberg

with $\mathbb{C}^+$ denoting the open right half-plane. We will discuss (3.2) again in Section 5 when we discuss how to obtain appropriate polynomials q.

With $q(A)$ approximating $A^{-1/2}$, the matrix $A(q(A))^2$ should, in a loose sense, be closer to the identity than A is and should thus have a small condition number. This is in turn an indication that we will require fewer Arnoldi iterations. To be specific, if, e.g., A is Hermitian and positive definite, then the analysis in [28, Theorem 4.3] shows that the error of the mth Arnoldi approximation is bounded by an expression of the form $C\left((\sqrt{\kappa}-1)/(\sqrt{\kappa}+1)\right)^m$, where κ is the condition number of the matrix. See Section 3.2 below for a more in-depth investigation of the condition number of the polynomially preconditioned matrix.

3.1. Algorithmic aspects. Algorithm 1 describes the left polynomially preconditioned Arnoldi method to approximate $A^{-1/2}b$ in detail. The Arnoldi process produces the orthonormal basis $v_1, \ldots, v_m$ of $\mathcal{K}_m((q(A))^2A, q(A)b)$ obeying the Arnoldi relation

$$(q(A))^2AV_m = V_mH_m + h_{m+1,m}v_{m+1}e_m^{\mathsf{T}},$$

with e_m the mth canonical unit vector in $\mathbb{C}^m$. The vector $H_m^{-1/2}e_1$ for the small matrix H_m is computed using an appropriate method for dense matrices, for example by computing $H_m^{1/2}$ via the blocked Schur algorithm [19] and then solving a linear system. In line 4 we explicitly stress that we compute the preconditioned matrix-vector product in three stages as multiplications with A and two times $q(A)$. This avoids numerically computing a representation of $p(z) = (q(z))^2$; see Section 5 for further discussion.

Algorithm 2 gives the details for right preconditioning. We changed the order of factors in line 4, which allows us to store the preconditioned vectors $y_j = q(A)v_j$ and use them when computing f_m. In this manner, right preconditioning saves one matrix-vector multiplication with $q(A)$ as compared to left preconditioning, where we cannot avoid computing $q(A)b$; see line 2 in Algorithm 1. If we do not store the y_j in Algorithm 2, then the approximation f_m must be computed as $q(A)(V_m(H_m^{-1/2}e_1\|b\|))$, which needs an additional (matrix-vector) multiplication with $q(A)$ for all those m for which we wish to compute this approximation.

With left preconditioning, the norm $\|f_m\|$ (and by extension, also the norms of the differences $\|f_m - f_{m+k}\|$, $k \geq 1$) can be obtained just from $H_m^{-1/2}e_1\|b\|$ (and $H_{m+k}^{-1/2}e_1\|b\|$), since V_m is orthonormal. This is not the case with right preconditioning, where Y_m does not have orthonormal columns.

The matrices $(q(A))^2A$ and $A(q(A))^2$ are identical, which is why we can expect that left and right preconditioning require about the same number of iterations for a given target accuracy, and this is what we observed in numerical tests not reported here. In conclusion, thus,

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Algorithm 2 m steps of right polynomially preconditioned Arnoldi for $A^{-1/2}b$.

choose polynomial q such that $q(A)$ approximates $A^{-1/2}$
put $v_1 \leftarrow b/\|b\|$
for $j = 1, \ldots, m$ **do** ▷ Arnoldi process
  compute $y_j \leftarrow q(A)v_j$, $u \leftarrow q(A)y_j$, $w \leftarrow Au$
  for $i = 1, \ldots, j$ **do**
    $h_{ij} \leftarrow \langle w, v_i \rangle$, $w = w - v_i h_{ij}$ ▷ orthogonalize against previous vectors
  end for
  $h_{j+1,j} \leftarrow \|w\|$
  $v_{j+1} \leftarrow w/h_{j+1,j}$
end for
$f_m \leftarrow Y_m(H_m^{-1/2}e_1\|b\|)$ ▷ $Y_m = [y_1|\cdots|y_m]$, $H_m = (h_{ij}) \in \mathbb{C}^{m\times m}$ upper Hessenberg

right preconditioning has the (minor) advantage of saving one matrix-vector multiplication with $q(A)$ if we can afford the additional storage for the vectors y_j. On the other hand, when basing a stopping criterion on the size of the difference of consecutive iterates, left preconditioning allows to compute this criterion more cheaply than right preconditioning.

3.2. Effect of preconditioning on the spectrum. To investigate the effect of preconditioning on the spectrum of A, we restrict ourselves to Hermitian positive definite A since this is conceptually simpler than more general settings. The preconditioning polynomial q is constructed as an approximation of the inverse square root on the spectral interval of A, and clearly the quality of the preconditioner will depend on the quality of this polynomial approximation. As a measure of this quality, we will in the following assume that we have a bound of the form

$$\left|\frac{1}{\sqrt{z}} - q(z)\right| \le \delta(z), \qquad \text{for } z \in [\lambda_{\min}, \lambda_{\max}], \tag{3.3}$$

for the approximation error, where $\lambda_{\min}$ and $\lambda_{\max}$ denote the smallest and largest eigenvalue of A, respectively. For example, $\delta(z) \equiv \varepsilon$ corresponds to a uniform bound for the *absolute* approximation error on the spectral interval, while $\delta(z) = \varepsilon/\sqrt{z}$ corresponds to a uniform bound for the *relative* approximation error.

If we have a uniform relative error bound available, then we can easily find an upper bound for the condition number of the preconditioned matrix $A(q(A))^2$.

PROPOSITION 3.1. *Let A be Hermitian positive definite with smallest and largest eigenvalue $\lambda_{\min}$ and $\lambda_{\max}$, respectively. Further, assume that we have a bound (3.3) with $\delta(z) = \varepsilon/\sqrt{z}$ available, where $\varepsilon < \sqrt{2} - 1 \approx 0.4142$. Then,*

$$\kappa_{pre} \le \frac{1 + 2\varepsilon + \varepsilon^2}{1 - 2\varepsilon - \varepsilon^2},$$

where κ_{pre} denotes the condition number of $A(q(A))^2$.

Proof. By a direct calculation, equation (3.3) implies that

$$\begin{aligned}
\left|1 - z(q(z))^2\right| &= \left|z\left(\frac{1}{z} - (q(z))^2\right)\right| = z\left|\frac{1}{\sqrt{z}} - q(z)\right|\left|\frac{1}{\sqrt{z}} + q(z)\right| \\
&= z\left|\frac{1}{\sqrt{z}} - q(z)\right|\left|\frac{2}{\sqrt{z}} - \left(\frac{1}{\sqrt{z}} - q(z)\right)\right| \le z\delta(z)\left(\frac{2}{\sqrt{z}} + \left|\frac{1}{\sqrt{z}} - q(z)\right|\right) \\
&\le 2\sqrt{z}\delta(z) + z(\delta(z))^2
\end{aligned}$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

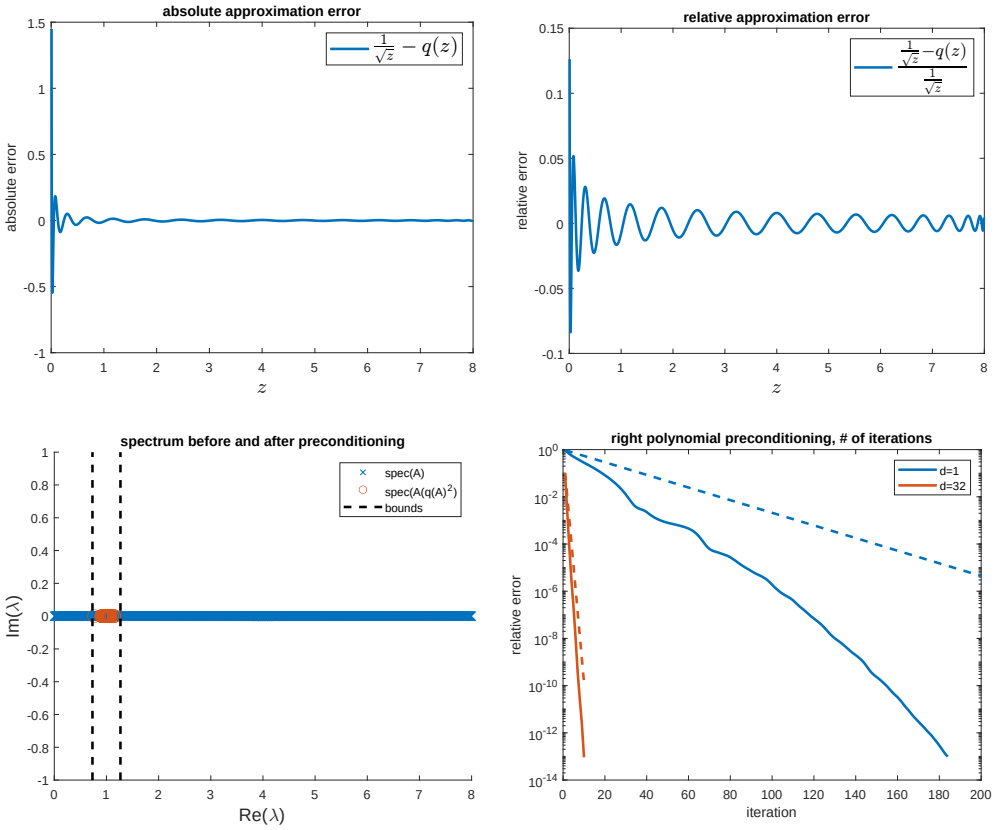

FIG. 3.1. *Illustration of effects of polynomial preconditioning for the discretized two-dimensional Laplace operator: Absolute/relative polynomial approximation error on the spectral interval (top left/right), effect on the spectrum (bottom left), convergence history and predicted slope (bottom right); see the text for details.*

for all $z \in [\lambda_{\min}, \lambda_{\max}]$. Inserting $\delta(z) = \varepsilon/\sqrt{z}$ gives

$$\max_{z\in[\lambda_{\min},\lambda_{\max}]} |1 - z(q(z))^2| \leq 2\varepsilon + \varepsilon^2. \tag{3.4}$$

Due to our assumption that $\varepsilon < \sqrt{2} - 1$, the right-hand side of (3.4) is smaller than one so that $z(q(z))^2$ takes values in $[1 - 2\varepsilon - \varepsilon^2, 1 + 2\varepsilon + \varepsilon^2]$, and the bound for the condition number directly follows. ∎

EXAMPLE 3.2. We illustrate our theory with a small example. Let $A \in \mathbb{R}^{2500\times 2500}$ be the discretization of the Laplace operator on a square with 50 interior grid points in each direction, with condition number $\kappa(A) \approx 1054$. We construct a Chebyshev preconditioning polynomial with $d = 32$ (see Section 5.1). The polynomial thus has degree 31.

The top left panel of Figure 3.1 displays the absolute difference between $q(z)$ and the inverse square root, while the top right panel displays the relative difference. Both in an absolute and in a relative sense, the largest errors occur towards the left end of the spectrum. A uniform relative error bound for this specific example is

$$\left| \frac{\frac{1}{\sqrt{z}} - q(z)}{\frac{1}{\sqrt{z}}} \right| \leq 0.1263 =: \varepsilon$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

so that according to Proposition 3.1, the condition number of $A(q(A))^2$ satisfies $\kappa_{\text{pre}} \leq 1.7345$. In the bottom left panel of Figure 3.1, we depict the actual eigenvalues of A and $A(q(A))^2$, together with the bounding interval $[1-2\varepsilon-\varepsilon^2, 1+2\varepsilon+\varepsilon^2]$. The actual condition number of $A(q(A))^2$ is 1.5153 and thus even a bit smaller than predicted by the bounds (and roughly a factor 700 smaller than the condition number of A). In the bottom right panel of Figure 3.1, we illustrate the convergence of the unpreconditioned and right preconditioned iteration together with the estimated convergence slopes based on the condition numbers of A and $A(q(A))^2$, respectively.

4. Polynomial preconditioning for the square root. For the square root, we could, in principle, take the same approach as for its inverse: We let q be a polynomial such that $q(A)$ again approximates $A^{-1/2}$ and then use

$$A^{1/2}b = (A(q(A))^2)^{1/2}(q^{-1}(A)b) = q^{-1}(A)(A(q(A))^2)^{1/2}b, \tag{4.1}$$

where $q^{-1}(z) = 1/q(z)$. Again, $A(q(A))^2$ will be much better conditioned than A for appropriate choices of q so that the Arnoldi approximations for its square root should converge rapidly. The new aspect is that we now also have to compute the action of $q^{-1}(A)$ on a vector. As the reciprocal of a polynomial, the function $q^{-1}(z)$ is a rational function, and its partial fraction expansion can be determined from the zeros of q. Thus, evaluating $q^{-1}(A)b$ means that we have to solve several shifted linear systems with A; just what we also have to do in a rational Krylov method.

REMARK 4.1. One can make the relation to rational Krylov methods more explicit: A general rational Krylov method extracts its iterates from a space of the form $\pi(A)^{-1}\mathcal{K}_k(A,b)$, where the denominator polynomial π is of degree $\leq k-1$; see, e.g., [34]. Performing m iterations of a polynomially preconditioned method based on (4.1) would correspond to using a denominator polynomial $\pi(z) = (q(z))^m$, i.e., cyclically repeat the same d poles, where d is the degree of q. Such a cyclic approach is actually common practice in many rational Krylov methods, as repeated poles allow to reuse matrix factorizations if a direct solver is employed [9, 34]. However, in "standard" implementations of rational Krylov methods, the polynomial part of the space is built with the same matrix with which shifted systems are solved. In contrast, a method based on (4.1) would build the polynomial part of the space via multiplications with $A(q(A))^2$, i.e., extract its approximation from $(q(A))^{-m}\mathcal{K}_m(A(q(A))^2, b)$. Such a method can thus be regarded as a polynomially preconditioned variant of the standard rational Krylov subspace method. The main computational work in rational Krylov methods lies in linear solves, and due to the typically small iteration numbers, orthogonalization cost and memory requirements are seldom an issue. Therefore, polynomial preconditioning appears to be less appealing in this setting.

Since our fundamental assumption was that efficient methods for solving shifted systems are not available, we consider a different approach instead. The idea is to use the identity $A^{1/2} = A^{-1/2}A$, which gives

$$A^{1/2}b = A^{-1/2}(Ab),$$

and then use preconditioning for the inverse square root. Note that the square root is also defined for a singular matrix provided its eigenvalue 0 is semi-simple, whereas the inverse square root is not. Interestingly, even then we can still proceed by using the (polynomially preconditioned) Arnoldi approximation for the inverse square root on Ab since the eigenvalue 0 is effectively deflated from the computation; a phenomenon termed *implicit desingularization* in [11]. The basis for this is the following theorem:

ETNA
Kent State University and
Johann Radon Institute (RICAM)

THEOREM 4.2. *Assume that $A \in \mathbb{C}^{n\times n}$ is singular and that its eigenvalue 0 is a k-fold semi-simple eigenvalue. Let*

$$A = TJT^{-1}, \quad \text{with } T \text{ nonsingular and } J = \oplus_{j=1}^{m} J_{n_j}(\lambda_j),$$

be its Jordan canonical form with Jordan blocks

$$J_{n_j} = \begin{bmatrix} \lambda_j & 1 & 0 & \cdots & 0 \\ 0 & \lambda_j & 1 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \lambda_j & 1 \\ 0 & \cdots & \cdots & 0 & \lambda_j \end{bmatrix} \in \mathbb{C}^{n_j\times n_j},$$

where $n = \sum_{j=1}^{m} n_j$. Assume that we order the eigenvalues by enumerating the zero eigenvalues first, i.e.,

$$\lambda_1 = \ldots = \lambda_k = 0 \quad \text{and} \quad n_1 = \ldots = n_k = 1, \quad \lambda_j \neq 0, \ \text{for } j = k+1, \ldots, m.$$

Denote by t_j the columns of $T = [t_1|\cdots|t_n]$, put $T_{\neg k} = [t_{k+1}|\cdots|t_n] \in \mathbb{C}^{n\times(n-k)}$, and let V be the subspace of $\mathbb{C}^n$ spanned by $t_{k+1}, \ldots, t_n$. Furthermore, let $\widehat{A}$ denote the restriction of A onto V, defined by

$$\widehat{A} : V \to V, \quad \widehat{A}t_j = At_j, \quad j = k+1, \ldots, n.$$

Then $\widehat{A}$, as a linear map from V to V, is nonsingular and

$$A^{1/2}y = \widehat{A}^{1/2}y \quad \text{for all } y \in V.$$

Proof. As a mapping from V to V, the Jordan blocks of $\widehat{A}$ are the blocks $J_{n_j}(\lambda_j)$, $j = k+1, \ldots, m$, and $t_{k+1}, \ldots, t_n$ are the corresponding generalized eigenvectors. This shows that $\widehat{A}$ is nonsingular. Moreover, the matrix functions $A^{1/2}$ and $\widehat{A}^{1/2}$ can be characterized using their Jordan canonical forms by their action on the generalized eigenvectors via

$$A^{1/2}T = TJ^{1/2}, \qquad \text{with} \quad J^{1/2} = \oplus_{j=1}^{m}\big(J_{n_j}(\lambda_j)\big)^{1/2},$$
$$\widehat{A}^{1/2}T_{\neg k} = T_{\neg k}\widehat{J}^{1/2}, \quad \text{with} \quad \widehat{J}^{1/2} = \oplus_{j=k+1}^{m}\big(J_{n_j}(\lambda_j)\big)^{1/2}.$$

So, if $y = \sum_{j=k+1}^{n} \eta_j t_j$, then with $a = [0\cdots 0\ \eta_{k+1}\cdots\eta_n]^{\mathsf{T}}$ we have

$$A^{1/2}y = A^{1/2}Ta = TJ^{1/2}a = T\begin{bmatrix} 0_k \\ \oplus_{j=k+1}^{m}\big(J_{n_j}(\lambda_j)\big)^{1/2}[\eta_{k+1}\cdots\eta_n]^{\mathsf{T}} \end{bmatrix}$$
$$= T_{\neg k}\widehat{J}^{1/2}\begin{bmatrix}\eta_{k+1}\\ \vdots \\ \eta_n\end{bmatrix} = \widehat{A}^{1/2}T_{\neg k}\begin{bmatrix}\eta_{k+1}\\ \vdots \\ \eta_n\end{bmatrix} = \widehat{A}^{1/2}y. \qquad \square$$

If $b \in \mathbb{C}^n$ is an arbitrary vector and 0 is a semi-simple eigenvalue of A, then the contributions of eigenvectors belonging to this eigenvalue in $y = Ab$ are deflated, i.e., y is of the form assumed in Theorem 4.2. Moreover then, using the notation of Theorem 4.2, we have that for any polynomial q, the preconditioned Krylov subspace $\mathcal{K}_m(A(q(A))^2, Ab)$ is the direct sum of $\mathcal{K}_m(\widehat{A}(q(\widehat{A}))^2, \widehat{A}b)$ and k times the space 0. Practically, this means that we can now choose q such that $q(\widehat{A})$ approximates $\widehat{A}^{-1/2}$ with $\widehat{A}$ nonsingular, and that we can compute Arnoldi approximations for $\widehat{A}^{-1/2}(\widehat{A}b)$ by working with $\mathcal{K}_m(A(q(A))^2, Ab)$ without any modifications at all.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

5. Choice and evaluation of the polynomial. We now discuss several ways to determine a suitable preconditioning polynomial q, i.e., a polynomial of a given degree for which $q(A)$ is a good approximation of $A^{-1/2}$. Of course, the difficulty of this task and its solution depend crucially on properties of the matrix A.

An important question is how far we can guarantee or at least expect that the relation (3.1) is fulfilled for the polynomial that we choose, i.e., whether we have $((q(A))^2)^{1/2} = q(A)$. In this respect, we can state the following general result:

THEOREM 5.1. *Assume spec*$(A) \subseteq \mathbb{C}^+$ *and that* q *approximates* $z^{-1/2}$ *on spec*(A) *uniformly in a relative sense with accuracy* $\frac{1}{\sqrt{2}}$*, i.e., we have*

$$|q(\lambda) - \lambda^{-1/2}| \leq \frac{1}{\sqrt{2}} |\lambda^{-1/2}|, \qquad \text{for } \lambda \in \text{spec}(A). \tag{5.1}$$

Then $((q(A))^2)^{1/2} = q(A)$.

Proof. Since $\text{spec}(A) \subseteq \mathbb{C}^+$, we have $|\arg(\lambda^{-1/2})| \leq \frac{\pi}{4}$ for all $\lambda \in \text{spec}(A)$. Because of (5.1), this implies

$$|\arg(q(\lambda))| < \frac{\pi}{2},$$

which in turn gives $|\arg((q(\lambda))^2)| < \pi$, and thus $((q(\lambda))^2)^{1/2} = q(\lambda)$ for all $\lambda \in \text{spec}(A)$, i.e., $((q(A))^2)^{1/2} = q(A)$. ∎

The assumption on the relative approximation error in (5.1) is quite restrictive. Heuristically, it seems justified to assume that for small values of $\lambda \in \mathbb{C}^+$, a polynomial q that approximates the inverse square root is such that $q(\lambda) \in \mathbb{C}^+$, too, since, after all, $\lambda^{-1/2} \in \mathbb{C}^+$ is large with $|\arg(\lambda^{1/2})| \leq \frac{\pi}{4}$. For large values of λ though, it might happen that $q(\lambda) \in \mathbb{C}^-$. If this is the case, then we do not have $((q(A))^2)^{1/2} = q(A)$, meaning that in the left preconditioned method, for example, we are starting with the "wrong" vector $c = q(A)b$ rather than the correct one $\widehat{c} = ((q(A))^2)^{1/2} b$. The expansions of c and $\widehat{c}$ in terms of the generalized eigenvectors of A differ in those eigenvectors that belong to the large eigenvalues λ for which $q(\lambda) \in \mathbb{C}^-$. This difference will not affect our computation of $A^{-1/2}c$ too much since the inverse square root effectively damps the components belonging to eigenvectors with large eigenvalues.

To summarize this discussion, we see that it is not trivial to guarantee that we have $((q(A))^2)^{1/2} = q(A)$. In special situations we might be able to actually assert this with mathematical certainty. In many other cases we might have strong heuristics or numerical indications that this identity holds. One such situation is when $\text{spec}(A) \subset \mathbb{C}^+$, where then $1/\sqrt{\lambda}$ is in the wedge $\{z \in \mathbb{C}^+, |\arg(z)| \leq \pi/4\}$, so that we can expect $q(\lambda) \in \mathbb{C}^+$ at least for all eigenvalues λ which are not too large. Moreover, if we do not aim for particularly high accuracy, then we can accept that for some of the large eigenvalues λ we violate $((q(\lambda))^2)^{1/2} = q(\lambda)$ when computing the action of the inverse square root. In any case, i.e., even when $((q(A))^2)^{1/2} \neq q(A)$, left or right polynomial preconditioning always yields an inverse square root which, though, might be non-primary: With left preconditioning we obtain $f = Bb$ with $B = (A(q(A))^2)^{-1/2}((q(A))^2)^{1/2}$, which satisfies $B^2 = A^{-1}$, and similarly for right preconditioning. We refer to [40] for further considerations on non-primary, primary, and principal (inverse) square roots.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

5.1. Chebyshev expansions. A real function $f : [\underline{a}, \overline{a}] \subseteq \mathbb{R} \to \mathbb{R}$ which is absolutely integrable with respect to the weight function $w(x) = ((x - \underline{a})(\overline{a} - x))^{-1/2}$ gives rise to a truncated Chebyshev series expansion

$$\sum_{i=0}^{k} c_i T_i^{[\underline{a},\overline{a}]}(z),$$

where $T_i^{[\underline{a},\overline{a}]}$ is the (scaled) Chebyshev polynomial of the first kind of degree i for the interval $[\underline{a}, \overline{a}]$, and the coefficients c_i are obtained as

$$c_0 = \frac{1}{\pi} \int_{\underline{a}}^{\overline{a}} w(z) \cdot f(z) T_0^{[\underline{a},\overline{a}]}(z)\, dz, \quad c_i = \frac{2}{\pi} \int_{\underline{a}}^{\overline{a}} w(z) \cdot f(z) T_i^{[\underline{a},\overline{a}]}(z)\, dz,\ i = 1, 2, \ldots$$

So, if the spectrum of A is enclosed in the real interval $[\underline{a}, \overline{a}] \subset (0, \infty)$—which is the case in particular if A is Hermitian and positive definite—we can, for a given degree, take q as the corresponding truncated Chebyshev expansion for $z^{-1/2}$ on $[\underline{a}, \overline{a}]$. The theory and the practical implementation of Chebyshev expansions are well studied and understood; see, e.g., [14]. In particular, we can use a straightforward matrix-vector version of the Clenshaw recurrence [16, 26] to accurately compute the action of $q(A)$ on a vector b.

If $\operatorname{spec}(A) \subseteq [\underline{a}, \overline{a}]$, with $\underline{a} > 0$, then the relation $((q(A))^2)^{1/2} = q(A)$, which we need in order to obtain the principal square root (see (3.1)), is fulfilled for the truncated Chebyshev expansion polynomial q if q is positive on $[\underline{a}, \overline{a}]$. We can verify this at least numerically by evaluating the polynomial on a discrete set of points in $[\underline{a}, \overline{a}]$.

5.2. Polynomials interpolating at (harmonic) Ritz values. Another way to obtain a preconditioning polynomial is by using the Arnoldi method itself. The Ritz values, i.e., the eigenvalues of the upper Hessenberg matrix H_d arising from d steps of the Arnoldi process, can be used as approximations for the eigenvalues of A (and they tend to first approximate exterior eigenvalues well); see, e.g. [62, Section 6].

It is therefore a natural idea to choose the preconditioning polynomial q in our method as the polynomial of degree $d - 1$ that interpolates $1/\sqrt{z}$ at the d Ritz values. This has the attractive feature that it does not require any *a priori* knowledge about the spectral region of A but rather adapts itself automatically to the spectrum of A. This approach comes at the cost of d additional Arnoldi steps, i.e., d matrix-vector products and $d(d-1)/2$ inner products. The Arnoldi process for constructing H_d can either be started with the vector b or with a randomly drawn vector, which might sometimes be preferable; see [48, Section 3.6] for a discussion of this topic in the context of polynomial preconditioning for linear systems.

Alternatively, one might want to use *harmonic Ritz values* as interpolation points. These are the inverses of the Ritz values of A^{-1} with respect to the space $A\mathcal{K}_d(A, b)$ and can be computed as the eigenvalues of

$$\widetilde{H}_d := H_d + h_{d+1,d}^2 H_d^{-*} e_d e_d^*.$$

The inverse μ^{-1} of a harmonic Ritz value μ is contained in the field of values $\mathcal{F}(A^{-1}) \subset \mathbb{C}$ of A^{-1} [63, Section 5.1], and thus $|\mu| > 1/\max\{|z| : z \in \mathcal{F}(A^{-1})\}$. This shows that for a nonsingular matrix A, the harmonic Ritz values are bounded away from 0, while standard Ritz values can become 0 (or arbitrarily close to 0) if $0 \in \mathcal{F}(A)$. It is this property which often makes harmonic Ritz values more attractive than standard Ritz values when the function has a pole at 0. Using preconditioning polynomials based on (harmonic) Ritz-value information has recently received quite a lot of attention in the context of solving linear systems [46, 47, 48, 49] and eigenvalue computations [24].

ETNA
Kent State University and
Johann Radon Institute (RICAM)

If the field of values $\mathcal{F}(A)$ of A is in $\mathbb{C}^+$, then so are its spectrum and its standard as well as its harmonic Ritz values. So we know that the constructed polynomial q satisfies $((q(\mu))^2)^{1/2} = q(\mu) = \mu^{-1/2}$ for the standard or harmonic Ritz values μ, and we can interpret this as an indication that we may expect that indeed $((q(\lambda))^2)^{1/2} = q(\lambda)$ for $\lambda \in \mathrm{spec}(A)$.

When working with interpolating polynomials, care has to be taken to use a representation which is favorable numerically. Following [52] and [46, 47, 48, 49], we represent in our numerical experiments the polynomial in its Newton form based on a Leja ordering of the Ritz values and evaluate it with a Horner-type scheme.

5.3. Polynomials obtained via error minimization. A more involved way to obtain a preconditioning polynomial is adapting the strategy presented in [70] for GMRES. Let Γ be a contour, and assume that its interior contains the spectrum of A or at least the major part of it. Such Γ may be available due to *a priori* information or it can be constructed as a polygon from the Ritz values of the Arnoldi process; see [70]. Note that A allows for a canonical (the "principal") inverse square root $A^{-1/2}$ if it has no eigenvalues on $(-\infty, 0]$. Thus, we can further assume that Γ and its interior exclude $(-\infty, 0]$.

If we discretize Γ into a set of points $\omega = \{z_1, \ldots, z_N\}$, then an inner product for the space $\mathcal{P}_{d-1}$ of polynomials of degree at most $d-1 < N$ is given by

$$\langle p_1, p_2 \rangle_\omega = \sum_{i=1}^{N} p_1(z_i)\overline{p_2(z_i)} \tag{5.2}$$

with corresponding norm $\|\cdot\|_\omega$. Here the bar denotes complex conjugation. We obtain the preconditioning polynomial $q(z)$ by solving the least-squares problem (where we extend $\|\cdot\|_\omega$ to functions other than polynomials)

$$\min_{q \in \mathcal{P}_{d-1}} \|z^{-1/2} - q(z)\|_\omega.$$

The theoretical justification for this approach is that if the discretization is fine enough, then the absolute value of the approximation error $z^{-1/2} - q(z)$ is small on all of the boundary Γ. As this error is holomorphic (note that we excluded $(-\infty, 0]$ from the interior of Γ), the maximum modulus principle [59, Theorem 10.24] states that $|z^{-1/2} - q(z)|$ cannot attain a larger value in the interior of Γ. Since Γ is constructed such that it approximately encloses the eigenvalues of A, $q(z)$ should thus yield a good approximation of $z^{-1/2}$ for all eigenvalues of A.

Because of the pole of $z^{-1/2}$ at 0, it becomes increasingly difficult to acquire a polynomial with small approximation error the closer Γ is to 0. Thus, it can be helpful to require a minimum absolute value of Γ. We implement this by replacing the part of Γ closer to 0 than a user-specified minimum distance by the circular arc of corresponding radius.

As described in [70], the above choice for a polynomial inner product allows us to use the standard Arnoldi process to construct an implicit polynomial basis. For this, we represent a polynomial p by the unique vector $\mathrm{vec}(p)$ containing its values on $z_1, \ldots, z_N$, i.e., $\mathrm{vec}(p) = [p(z_1), \ldots, p(z_N)]^\mathsf{T}$. Then the inner product (5.2) corresponds to the standard Euclidean inner product for these vector representations. If we apply the Arnoldi process to the matrix $\mathrm{diag}(z_1, \ldots, z_N)$ and the vector $[1, \ldots, 1]^\mathsf{T}/N^{1/2} \in \mathbb{R}^N$, then the kth Arnoldi vector represents the kth orthonormal basis polynomial $p_{k-1}(z)$ (with respect to (5.2)) of degree $k-1$. We express $q(z)$ in terms of these basis polynomials, so the least-squares problem

$$\min_{q \in \mathcal{P}_{d-1}} \|z^{-1/2} - q(z)\|_\omega = \min_{\alpha_j \in \mathbb{C}} \|z^{-1/2} - \sum_{j=0}^{d-1} \alpha_j p_j(z)\|_\omega$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

is equivalent to

$$\min_{\alpha_j \in \mathbb{C}} \| \operatorname{vec}(z^{-1/2}) - \sum_{j=0}^{d-1} \alpha_j \operatorname{vec}(p_j) \|_2 = \min_{\alpha \in \mathbb{C}^d} \| \operatorname{vec}(z^{-1/2}) - P_d \alpha \|_2, \tag{5.3}$$

where the matrix $P_d = [\operatorname{vec}(p_0) | \dots | \operatorname{vec}(p_{d-1})] \in \mathbb{C}^{N \times d}$ contains the Arnoldi vectors as its columns. Since they are orthonormal, the vector α with components α_i is obtained as $\alpha = P_d^* \operatorname{vec}(z^{-1/2})$. Once α is known, we can evaluate $q(A)v$ using an Arnoldi-like process that does not require knowledge of P_d; see [70, Section 3] for more information. This process involves $d-1$ matrix-vector products with A and $\mathcal{O}(d^2)$ vector operations but no inner products.

Within this approach, we know the value of $q(z)$ on discrete points on the contour Γ, which (approximately) contains the spectrum of A. If then $q(z) \in \mathbb{C}^+$ for these points, then this can be taken as an indication that indeed $q(\lambda) \in \mathbb{C}^+$ for all $\lambda \in \operatorname{spec}(A)$ as is required in (3.1): If $q(z) \in \mathbb{C}^+$, i.e., $\operatorname{Re}(q(z)) > 0$ not only for the discrete points on the contour Γ but on all of the contour, then $\operatorname{Re}(q(z)) > 0$ for all z inside the contour since the real part of the holomorphic function q is harmonic and thus attains its minimum for all z inside Γ on the boundary Γ; see Section 15.1 and in particular Theorem 15.1g in [38].

We note that we could actually add $\operatorname{Re}(\operatorname{vec}(q)) \geq 0$ as a constraint to (5.3). This constraint can be written as

$$0 \leq \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} \operatorname{Re}(\operatorname{vec}(q)) \\ \operatorname{Im}(\operatorname{vec}(q)) \end{bmatrix} = \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} \operatorname{Re}(P_{d+1}) & -\operatorname{Im}(P_{d+1}) \\ \operatorname{Im}(P_{d+1}) & \operatorname{Re}(P_{d+1}) \end{bmatrix} \begin{bmatrix} \operatorname{Re}(\alpha) \\ \operatorname{Im}(\alpha) \end{bmatrix}.$$

Furthermore, we can rewrite (5.3) in a similar manner, separating real and imaginary components so that in total we obtain a real inequality-constrained linear least-squares problem. These are known to be equivalent to convex quadratic programming problems, for which several efficient algorithms exist; see, e.g., [12, Section 5.2.2] and the references therein. In the following examples this will not be necessary however, as the constraint will be satisfied without having been required explicitly in the optimization problem (5.3).

6. Numerical examples. We now illustrate the benefits of polynomial preconditioning for several examples. The experiments in Examples 6.1 and 6.3 are run in MATLAB R2022a on a computer with Intel Core i7-1185G7 8-core CPU (3.0 GHz) and 32 GB RAM under Ubuntu 20.04, while Example 6.2 has been implemented in parallel via `C` and `MPI` and was run on the JUWELS Cluster of the Jülich Supercomputing Centre. Each node of JUWELS consists of two Intel Xeon Platinum 8168 CPUs running at 2.7 GHz and with 96 GB of RAM; see [44] for further information.

EXAMPLE 6.1 (Inverse square root for the 3D discrete Laplace operator).

In our first example we take $A \in \mathbb{R}^{n \times n}, n = 10^6$, as the discrete, three-dimensional Laplace operator on a grid with $100 \times 100 \times 100$ equidistant interior points and choose b as a vector with normally distributed random entries, scaled to have norm 1. The spectral interval of A is $[6(1-\cos(\frac{\pi}{N+1})), 12-6(1-\cos(\frac{\pi}{N+1}))]$, and we obtained preconditioning polynomials via the Chebyshev expansion over the spectral interval as described in Section 5.1. We (numerically) verified that $q_{d-1}(z)$, for the various degrees $d-1$ used, does not have a zero on the spectral interval $[\underline{a}, \overline{a}]$ and thus maps $[\underline{a}, \overline{a}]$ onto a positive real interval. This means that by (3.2) we indeed have $((q(A))^2)^{1/2} = q(A)$.

The top row of Figure 6.1 depicts the relative 2-norm $\|f_m - A^{-1/2}b\|/\|A^{-1/2}b\|$ of the error of the Arnoldi approximation f_m as a function of the iteration number m; see Algorithm 2. The "exact" solution $A^{-1/2}b$ that we use to compute the errors is actually an

ETNA
Kent State University and
Johann Radon Institute (RICAM)

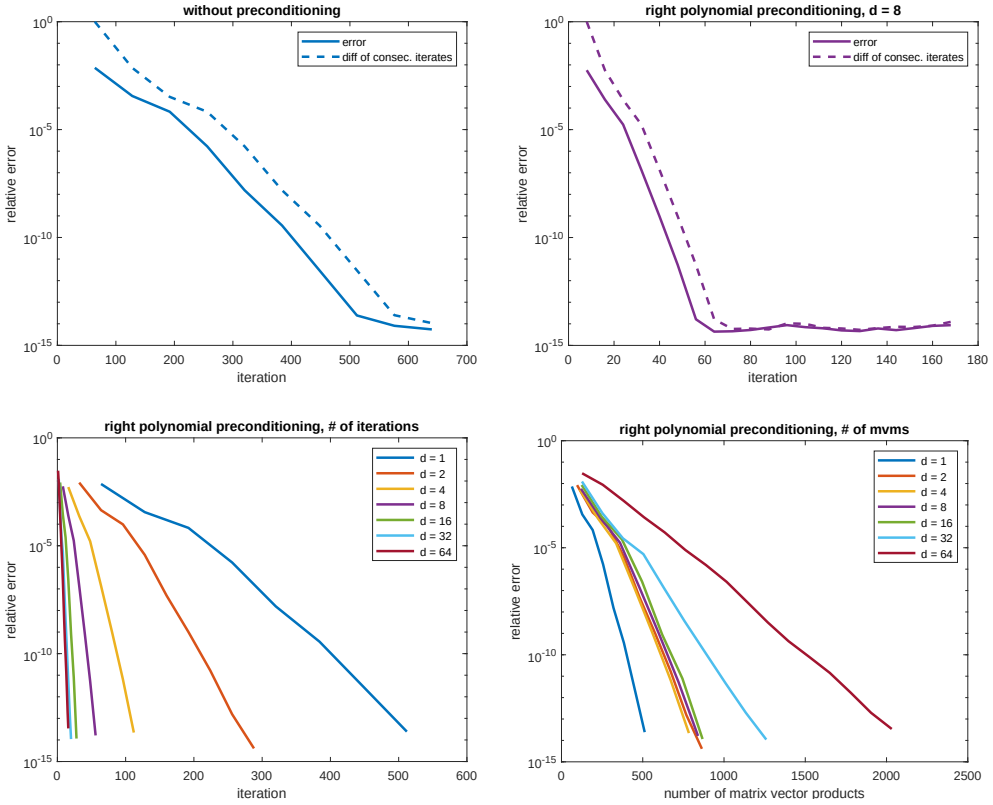

FIG. 6.1. *Relative error when approximating $A^{-1/2}b$ with Chebyshev preconditioning polynomials of various degrees $d-1$ ($d=1$ corresponds to an unpreconditioned method), where A is the discretization of the three-dimensional Laplace operator and b is a vector with normally distributed random entries, scaled to have norm 1.*

approximation computed using a restarted Arnoldi iteration with guaranteed error bounds as described in [31] with the bound set to 10^{-14}.

The left plot in Figure 6.1 is without preconditioning, and the right plot is for $d=8$, i.e., we use a degree-7 polynomial as a right preconditioner. Both plots also show the relative 2-norm $\|f_{m+k}-f_m\|/\|f_{m+k}\|$ of the difference of two iterates (as dashed lines), a quantity that is easily available also for large matrices—even without explicitly forming the iterates. Here, the parameter $k \geq 1$ controls how frequently this error estimate is evaluated (and thus, how frequently the inverse square root of the respective $m \times m$ Hessenberg matrix needs to be computed). The plots show that these norms of the differences start to stagnate at the same time when the errors start to stagnate, so this observation can be used to devise a simple stopping criterion. To account for the fact that iterations of the preconditioned method are much more costly (in particular for higher values of d) than iterations of the plain method but one expects that much fewer iterations are required for convergence, we make the parameter k smaller the larger d is. Specifically, in this experiment with $d=1,2,4,\ldots,64$, we choose $k=64/d$, so that we check the residual norm every 64th iteration in the plain method and every iteration when $d=64$.

The bottom part of Figure 6.1 shows results for various degrees $d-1$ of the preconditioning polynomial. The left plot depicts the error as a function of the number of Arnoldi iterations, while the right plot shows it as a function of the number of matrix-vector multiplications (mvms). Recall that preconditioning with a degree-$(d-1)$ polynomial requires $2d-1$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

TABLE 6.1
Timings and operation counts for approximating $A^{-1/2}b$ with Chebyshev preconditioning polynomials of various degrees, where A is the discretization of the three-dimensional Laplace operator and b is a vector with normally distributed random entries, scaled to have norm 1. The polynomial q has degree $d-1$, i.e., the first row corresponds to the plain, unpreconditioned Lanczos method. Run times are given both for the standard (1P) and two-pass (2P) version of the algorithm.

d	iterations	mvms	inner products	time 1P (in s)	time 2P (in s)
1	512	512	1 024	8.0	14.5
2	288	864	576	12.9	25.8
4	112	784	224	10.7	20.4
8	56	840	112	12.2	20.7
16	28	868	56	10.8	22.8
32	20	1 260	40	16.9	32.6
64	16	2 032	32	24.3	53.9

matrix-vector multiplications per iteration. We can take this plot as an information about the computational cost, since mvms are the by far most costly operations here. We observe that the preconditioned Arnoldi methods take increasingly more mvms as d is increased, and they all take more mvms than the standard method. However, if storage were an issue and we could not store the 512 Arnoldi vectors, then the standard method could not be performed as such but rather as a two-pass Lanczos approach, where the first part does not store the basis vectors but just assembles the tridiagonal matrix H_m (see, e.g., [32, 36]). Once the coefficient vector $H_m^{-1/2}e_1\|b\|$ is computed, a second pass is done which then combines the anew computed basis vectors using these coefficients. In this way, the number of mvms is actually twice as much—1 024 in our example—as what we see in the plot. Then, if for example we can store the about 110 vectors needed for preconditioning with $d = 4$, then the preconditioned method with its 784 mvms takes 24% less mvms than the standard method.

In Table 6.1 we report the wall clock time for running the plain and preconditioned Lanczos method, both in a one-pass and two-pass version, together with the number of iterations, mvms, and inner products needed to reach a relative error norm below 10^{-12}. Among all tested methods, the plain (unpreconditioned) Lanczos method has the smallest run time, followed by polynomial preconditioning with $d = 4$. As explained in the previous paragraph, for very large-scale problems, it might actually be relevant to compare the plain *two*-pass Lanczos method with that of the preconditioned *one*-pass Lanczos, as the latter method builds a much smaller subspace and is thus less likely to require two passes due to memory constraints. We therefore note that for all values of d between 2 and 16, the run time of the one-pass preconditioned method is lower than that of the two-pass unpreconditioned method.

EXAMPLE 6.2 (Sign function for the overlap operator in lattice QCD). For a square matrix A, its sign function $\mathrm{sign}(A)$ can be expressed as $A(A^2)^{-1/2}$, so that the computational burden in computing $\mathrm{sign}(A)b$ resides in computing $(A^2)^{-1/2}b$. Note that A^2 will not be computed explicitly; we rather compute A^2x for a given vector x as two consecutive mvms with A.

The sign function of a large, non-Hermitian matrix arises in the overlap operator in lattice Quantum Chromodynamics (QCD) as we now shortly explain. QCD is the fundamental physical theory of the quarks and gluons as the constituents of matter, and their interaction via the strong force is described by the Dirac operator. The Wilson-Dirac operator D_w arises as a discretization of the Dirac operator on a finite four-dimensional Cartesian lattice. The Wilson-Dirac operator acts on discrete spinor fields which have twelve components per grid point, corresponding to all possible combinations of three color and four spin indices [33].

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Now, $D_{ov}(\mu)$, the *overlap Dirac operator* at the chemical potential μ, preserves chiral symmetry, an important physical property, on the lattice while other discretizations as, e.g., D_w do not. To be specific, the overlap Dirac operator takes the form [13, 39, 54]

$$D_{ov}(\mu) = I + \rho\Gamma_5 \text{sign}(\underbrace{\Gamma_5 D_w(m_w,\mu)}_{=:Q(m_w,\mu)}).$$

Here, Γ_5 is a simple diagonal matrix which acts as the identity on spinor components belonging to spins 1 and 2 and as the negative identity on those belonging to spins 3 and 4, and $\rho \in (0,1)$ is a mass parameter, typically close to 1. In the argument of the sign function, $D_w(m_w,\mu)$ is the massless Dirac-Wilson operator with an appropriately chosen shift $m_w \in (-2,0)$ and a chemical potential of μ. It is the presence of $\mu \neq 0$ that makes $Q(m_w,\mu)$ non-Hermitian; see [13]. For notational simplicity, we abbreviate $Q(m_w,\mu)$ as Q_μ from now on.

In our computations, we use a Dirac matrix for a grid with dimensions 64×32^3 coming from a physically relevant ensemble provided by the lattice QCD group at the University of Regensburg via the Collaborative Research Centre SFB-TRR55, with parameters $m_0 = -0.332159624413$ and $c_{sw} = 1.9192$ [8]. Hence, Q_μ has 25 165 824 rows and columns. We took $m_w = -1.4$ and $\mu = 0.3$, which are physically relevant values. All our results relate to the evaluation of $(Q_\mu^2)^{-1/2}b$, where b is a vector with uniformly distributed random entries. In an actual simulation in lattice QCD, one has to solve systems with $D_{ov}(\mu)$ repeatedly, using an iterative solver. Each iteration then requires an evaluation of $\text{sign}(Q_\mu)b$ for some vector b, and this in turn is obtained by computing $Q_\mu \cdot (Q_\mu^2)^{-1/2}b$.

We obtain the preconditioning polynomial q via interpolation at the Ritz values (of Q_μ^2) as outlined in Section 5.2. Since all Ritz values were always contained in the right half-plane, so were the values of $q(z) = z^{-1/2}$ at these Ritz values, which is important in view of (3.1). When using harmonic Ritz values instead of standard Ritz values, we consistently needed about 10% more iterations across all tested values of d. For this reason, we do not report results using harmonic Ritz values. For this example, we report results for left preconditioning with results for right preconditioning being virtually the same.

Our numerical experiments were run on 64 and 256 nodes on the JUWELS Cluster, and we use 2 `MPI` processes per node and 24 `OpenMP` threads per process.[2] As the "exact" solution f^*, we took the unpreconditioned f_{m+k} with $m = 6\,000$ and $k = 64$ as before, and we obtained the error measure $\|f_m - f_{m+k}\|_2/\|f_{m+k}\|_2 \approx 3.0 \cdot 10^{-10}$.

Computing $H_m^{-1/2}$ for the Hessenberg matrices H_m is done using the SLEPc library [58], with its cost in seconds as a function of m shown in the right plot of Figure 6.2. The time to obtain $H_m^{-1/2}$ was around five times larger than that for running the whole Arnoldi process with $m = 6\,000$ when running on 64 nodes on JUWELS, rendering the computation of $\|f_{m+k} - f_m\|_2/\|f_{m+k}\|_2$ ten times more expensive than the time for the Arnoldi method for that value of m. These times appear to be overly large, and we are convinced that they could be reduced substantially by looking at the details of the implementation. Rather than doing that, though, we decided to simply exclude the time needed for checking the stopping criterion from all the timings reported in Table 6.2. Note that the unpreconditioned method would have to check the stopping criterion more often and evaluate the sign function on larger Hessenberg matrices, so excluding this from the timings is in favor of the unpreconditioned method.

The unpreconditioned case converges quite slowly, so that running the Arnoldi method with full orthogonalization up to $m = 6\,000$ is certainly not realistic in practice because of the very high orthogonalization cost. For our comparison with the preconditioned methods, we therefore reduced the iteration number of the unpreconditioned Arnoldi method to 1 600

[2] code available at https://github.com/Gustavroot/sign_function_LQCD_with_polyprec

ETNA
Kent State University and
Johann Radon Institute (RICAM)

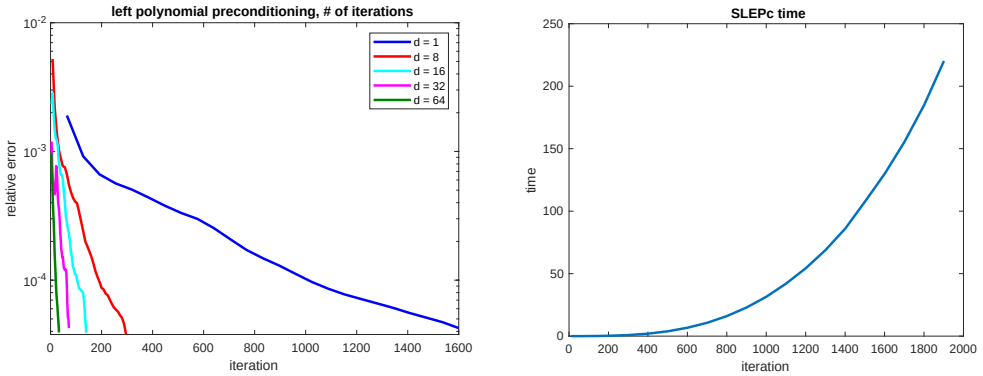

FIG. 6.2. *Results for approximating* $(Q_\mu)^2)^{-1/2}b$ *with Arnoldi-preconditioning polynomials of various degrees and* b *a uniformly distributed random vector. Left: relative error as a function of the Arnoldi basis size up to a value of* 1 600*. Right: time (in s) to compute* $H_m^{-1/2}$ *as a function of* m *with SLEPc.*

TABLE 6.2
Timings and operation counts for approximating $((Q_\mu)^2)^{-1/2}b$ *with Arnoldi-preconditioning polynomials* q *of various degrees* $d-1$ *and* b *a uniformly distributed random vector. The polynomial* q *has degree* $d-1$*, i.e., the first row corresponds to the plain, unpreconditioned Arnoldi method.*

d	iterations	mvms	inner products	time 64 nodes (in s)	time 256 nodes (in s)
1	1 600	3 200	1 279 200	127.8	105.8
8	296	8 910	42 510	25.7	9.8
16	140	8 742	9 991	12.3	7.8
32	72	9 198	3 125	11.6	7.4
64	33	8 636	2 578	10.6	5.5

iterations. This resulted in a relative error of around $4.0 \cdot 10^{-5}$ at $m = 1\,600$, which we then used as the stopping criterion in the preconditioned runs, too. The left part of Figure 6.2 displays the relative error as a function of the iteration counts, and Table 6.2 gives operations counts and timings. We checked the error every $k = 64/d$ iterations.

From Figure 6.2 and Table 6.2 we see that, as in the previous example, polynomial preconditioning reduces the number of iterations while increasing the total number of mvms. Still, the reduction in execution time is very pronounced for the preconditioned methods, because orthogonalization costs dominate when a large number of iterations have to be performed. The fastest preconditioned method arises for $d = 64$, and it is by a factor of twelve faster than the unpreconditioned method when using 64 nodes and by a factor of 19 on 256 nodes. In the additional Figure 6.3, we provide convergence plots for the best three preconditioned methods from Table 6.2, now going down to a relative error of 10^{-9}, a precision which is very difficult to attain for the unpreconditioned algorithm due to tremendous time and memory requirements. For this higher accuracy, $d = 32$ and $d = 64$ take almost the same time, while $d = 16$ takes about 10% more time since the orthogonalization cost, which grows quadratically in the number of iterations, starts to prevail. The case $d = 64$ with $m = 94$ is 135 times faster than the unpreconditioned one with $m = 6\,000$ on 256 nodes (which achieves comparable accuracy).

EXAMPLE 6.3 (Square root of graph Laplacian). As our last example we study another nonsymmetric problem, which is known to be notoriously difficult to solve both for sketched and for restarted Krylov subspace methods; see, e.g., [17, Section 5.1.3] and [37, Section 5.4], where a similar (but much smaller) model problem is considered. We let A be the adjacency

ETNA
Kent State University and
Johann Radon Institute (RICAM)

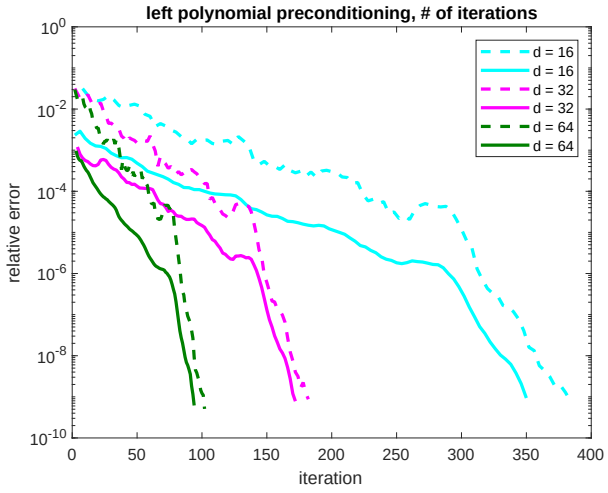

FIG. 6.3. *Results for approximating* $((Q_\mu)^2)^{-1/2}b$ *with Arnoldi-preconditioning polynomials of various degrees and* b *a uniformly distributed random vector. Solid lines: relative error* $\|f_m - f^*\|/\|f^*\|$*, dashed lines: error measure* $\|f_m - f_{m+k}\|/\|f_{m+k}\|$ *with* $k = 64/d$*.*

TABLE 6.3
Timings and operation counts for approximating $L^{1/2}b$ *with two choices for the preconditioning polynomials of various degrees* $d-1$*. Left: polynomials interpolate at the Ritz values (see Section 5.2). Right: error minimizing polynomials (see Section 5.3).* L *is the graph Laplacian of the network* `Kamvar/Stanford`*. The first row corresponds to the unpreconditioned Arnoldi method.*

	interpolation of Ritz values				error minimization			
d	iters	mvms	inprods	time (in s)	iters	mvms	inprods	time (in s)
1	928	929	860 256	333.0	928	929	860 256	333.0
2	576	1 732	331 202	141.7	544	1 693	298 932	393.9
4	312	2 190	97 044	58.1	272	1 965	77 252	125.3
8	176	2 650	30 856	37.7	144	2 221	24 132	67.4
16	102	3 180	10 542	35.1	78	2 479	9 546	60.7

matrix of a directed graph, D_{in} the diagonal matrix which contains the in-degrees of all nodes on the diagonal, and $L = D_{\text{in}} - A$ the corresponding *in-degree Laplacian*.

The square root $L^{1/2}$—a *fractional Laplacian*—is an important tool for modeling nonlocal dynamics on the network; see, e.g., [10, 50]. It is evident from its definition that all column sums of L are zero, and it is thus singular. Its spectrum is contained in the closed right half-plane as can be seen via Gershgorin's theorem, while its field of values is not; see [41, Theorem 1.6.6]).

We are interested in forming $L^{1/2}b$, where b is a randomly chosen canonical unit vector e_i. As outlined in Section 4, we apply our method to approximate $L^{-1/2}(Lb)$, which is possible despite L being singular; cf. Theorem 4.2. To increase numerical stability, we run the Arnoldi method with reorthogonalization in this example. The graph we took is `Kamvar/Stanford` from the SuiteSparse matrix collection [18], so L has 281 903 rows and columns and 2 594 228 nonzero entries.

In Figure 6.4 and the left part of Table 6.3 we report results obtained with right preconditioning, using polynomials interpolating at the Ritz values (see Section 5.2), this time aiming for a relative error of 10^{-7}. Given the large problem size, as in Example 6.1, we take as the

ETNA
Kent State University and
Johann Radon Institute (RICAM)

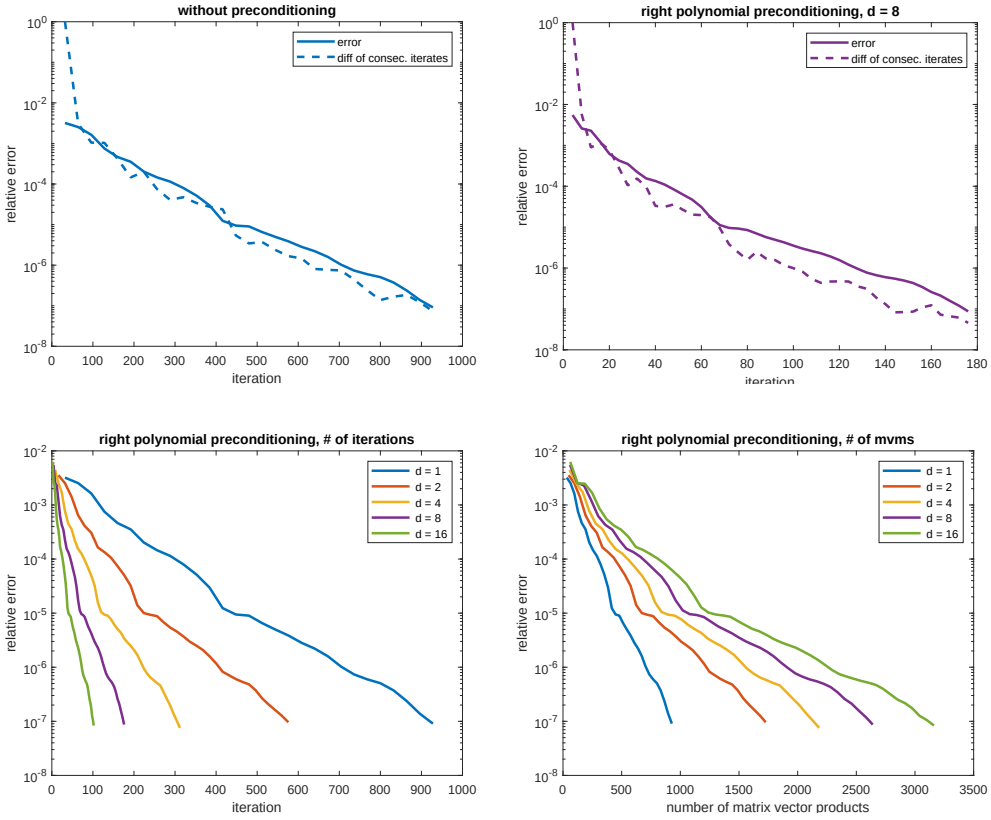

FIG. 6.4. *Results for approximating $L^{1/2}b$ with Arnoldi-preconditioning polynomials of various degrees ($d = 1$ corresponds to an unpreconditioned method), where L is the graph Laplacian of the network* `Kamvar/Stanford` *and b is a randomly chosen canonical unit vector e_i.*

"exact" solution, which we use to compute the errors, the approximation obtained with the unpreconditioned Arnoldi method with a stricter tolerance 10^{-10}. The parameter k that controls how frequently we check the stopping criterion is chosen as $k = 32/d$ in this experiment. As in the previous example, the benefits of polynomial preconditioning become very apparent: As orthogonalization cost largely dominates the overall cost of the unpreconditioned method, run time is reduced by a factor of about 9 when going from the unpreconditioned Arnoldi to a polynomially preconditioned Arnoldi method with $d = 8$ or $d = 16$. Note that Table 6.3 now reports complete timings, including the time spent in evaluating $H_m^{-1/2}b$ (via the function `sqrtm` of MATLAB followed by a linear system solve). As opposed to Example 6.2, the time spent in these evaluations now represents only a small fraction (from 0.5% for $d = 16$ to 2.8% for $d = 1$) of the total computing time.

The right part of Table 6.3 shows results obtained with preconditioning polynomials solving the least-squares problem discussed in Section 5.3. For this, we run 60 Arnoldi iterations and apply the MATLAB function `boundary` to the Ritz values to construct the boundary Γ. We choose a minimum absolute value of 0.1 for all of Γ and discretize the modified boundary in steps of uniform size 0.005. In view of (3.1) we tested whether for the discrete points z on the boundary we have $q(z) \in \mathbb{C}^+$, which was indeed always the case. We refrain from showing convergence plots here, as they do not give additional insight beyond what we see in Figure 6.4 for the Arnoldi polynomial.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

TABLE 6.4
Timings and operation counts for approximating $L^{1/2}b$ by sketched-and-truncated or restarted Krylov methods, where L is the graph Laplacian of the network `Kamvar/Stanford` *and b is a random vector of unit norm. For each class of methods, only the value of k leading to the most efficient outcome is reported in this table.*

method	trunc./rest. length k	mvms	inner products	time (in s)
sketched FOM	20	1 900	37 790	79.9
restarted Arnoldi	200	1 800	179 100	91.8
deflated restarted Arnoldi	100	1 400	69 300	60.9

The least-squares polynomials are complex and thus necessitate complex arithmetic even though L and b are real. This is in contrast to the interpolating polynomials which are real because all Ritz values are. As a consequence, with least-squares polynomials, the run time increases for $d = 2$ compared to the unpreconditioned case and remains larger than the run times with the interpolation polynomials for the same values of d. However, fewer matrix-vector and inner products are needed, so we expect this approach to be more beneficial in cases where complex arithmetic is needed anyways.

To conclude this example, we compare to the performance of competing methods that can be used to reduce storage and orthogonalization cost when approximating functions of nonsymmetric matrices. Specifically, we compare with the sketched FOM method from [37][3] as well as with quadrature-based restarting [29][4], both with and without implicit deflation [23]. The results are depicted in Figure 6.5.

For the restarted Arnoldi method, we vary the restart lengths from 20 to 200 and allow at most 3000 mvms overall. Without implicit deflation, the method only manages to reach the desired accuracy of 10^{-7} for the largest restart length $k = 200$, within nine restart cycles. This requires 1 800 mvms as well as 179 100 inner products and takes 91.8 seconds.

When using implicit deflation, we aim to deflate $k/10$ approximate eigenvectors (which is typically a suitable value). This speeds up convergence, particularly for $k = 50$ and $k = 100$. The overall most efficient method is then for $k = 100$, taking 60.9 seconds.

In sketched FOM, the Krylov basis is computed by a k-truncated orthogonalization, i.e., each new basis vector is only orthogonalized against the k preceding basis vectors (so that $k = 2$ mimics the Lanczos process). We try different truncation parameters ranging from 2 to 30. Several observations can be made: For the smaller truncation lengths, the method stagnates or diverges long before reaching the target accuracy. In general, convergence is very irregular and erratic, which also means that error estimates and stopping criteria can become unreliable. Additionally, the practical choice of a good truncation parameter appears to be non-trivial, as there seems to be no systematic dependence of performance on the truncation parameter (e.g., $k = 5$ performs better than $k = 10$ and $k = 20$ performs better than $k = 30$). The method performs best for $k = 20$ and manages to reach the desired accuracy within 1 900 iterations, requiring 79.9 seconds.

Overall, the preconditioned Arnoldi method with $d = 8$ or $d = 16$ outperforms the most efficient other method by a factor of roughly 2 in wall clock time.

7. Concluding remarks. We presented a way to use polynomial preconditioning in the Arnoldi method for computing the action of the inverse square root of a matrix on a vector, and by expansion, also for the square root itself. Due to the reduction of the number of Arnoldi steps, the polynomially preconditioned method saves on orthogonalization and storage cost, and these savings can be very substantial, particularly in the non-Hermitian case.

[3] available at https://github.com/MarcelSchweitzer/sketched_fAb.
[4] available at https://github.com/guettel/funm_quad.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

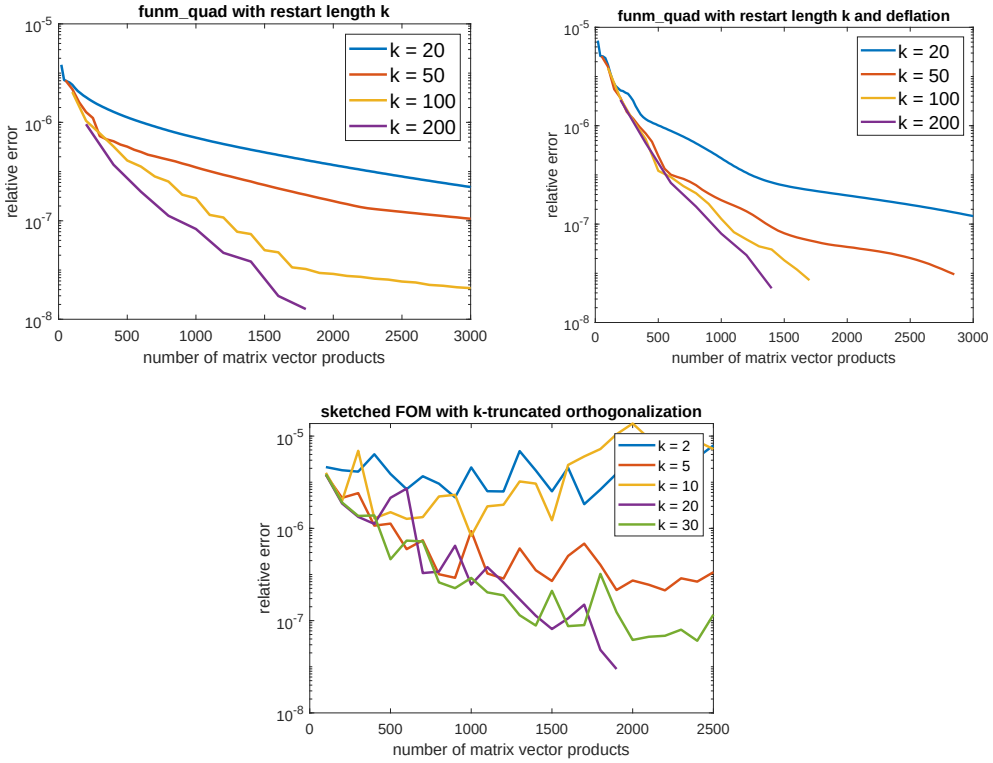

FIG. 6.5. *Results for approximating $L^{1/2}b$ by sketched-and-truncated or restarted Krylov methods with various truncation/restart lengths k, where L is the graph Laplacian of the network* `Kamvar/Stanford` *and b is a randomly chosen canonical unit vector.*

While an extension to inverse pth roots is possible, in principle, by using polynomials which are pth powers, polynomial preconditioning cannot be applied for general matrix functions. When it can be applied, though, it is conceptually and computationally simpler than other approaches such as sketching and restarting, which also aim at avoiding long recurrences. This is confirmed by our numerical examples which show that polynomial preconditioning may substantially outperform these other approaches. We also discussed why it is important to take into account that the square root has two branches.

Acknowledgement. The authors gratefully acknowledge the Gauss Centre for Supercomputing e.V. (`www.gauss-centre.eu`) for funding this project by providing computing time through the John von Neumann Institute for Computing (NIC) on the GCS Supercomputer JUWELS at Jülich Supercomputing Centre (JSC), under the project with id MUL-TRA. This work is partially supported by the German Research Foundation (DFG) research unit FOR5269 "Future methods for studying confined gluons in QCD".

REFERENCES

[1] M. AFANASJEW, M. EIERMANN, O. G. ERNST, AND S. GÜTTEL, *Implementation of a restarted Krylov subspace method for the evaluation of matrix functions*, Linear Algebra Appl., 429 (2008), pp. 2293–2314.
[2] H. ALZER AND C. BERG, *Some classes of completely monotonic functions*, Ann. Acad. Sci. Fenn. Math., 27 (2002), pp. 445–460.
[3] M. ARIOLI AND D. LOGHIN, *Discrete interpolation norms with applications*, SIAM J. Numer. Anal., 47 (2009), pp. 2924–2951.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

[4] S. F. ASHBY, *Polynomial Preconditioning for Conjugate Gradient Methods*, PhD. Thesis, University of Illinois at Urbana-Champaign, Champaign, 1988.
[5] O. BALABANOV AND L. GRIGORI, *Randomized block Gram–Schmidt process for solution of linear systems and eigenvalue problems*, Preprint on arXiv, 2021. https://arxiv.org/abs/2111.14641
[6] ———, *Randomized Gram-Schmidt process with application to GMRES*, SIAM J. Sci. Comput., 44 (2022), pp. A1450–A1474.
[7] O. BALABANOV AND A. NOUY, *Randomized linear algebra for model reduction. Part I: Galerkin methods and error estimation*, Adv. Comput. Math., 45 (2019), pp. 2969–3019.
[8] G. S. BALI, S. COLLINS, B. GLÄSSLE, M. GÖCKELER, J. NAJJAR, R. H. RÖDL, A. SCHÄFER, R. W. SCHIEL, A. STERNBECK, AND W. SÖLDNER, *The moment $\langle x \rangle_{u-d}$ of the nucleon from $n_f = 2$ lattice QCD down to nearly physical quark masses*, Phys. Rev. D, 90 (2014), Art. 074510, 9 pages.
[9] B. BECKERMANN AND L. REICHEL, *Error estimates and evaluation of matrix functions via the Faber transform*, SIAM J. Numer. Anal., 47 (2009), pp. 3849–3883.
[10] M. BENZI, D. BERTACCINI, F. DURASTANTE, AND I. SIMUNEC, *Non-local network dynamics via fractional graph Laplacians*, J. Complex Netw., 8 (2020), Art. cnaa017, 29 pages.
[11] M. BENZI AND I. SIMUNEC, *Rational Krylov methods for fractional diffusion problems on graphs*, BIT, 62 (2022), pp. 357–385.
[12] Å. BJÖRCK, *Numerical Methods for Least Squares Problems*, SIAM, Philadelphia, 1996.
[13] J. BLOCH AND T. WETTIG, *Overlap Dirac operator at nonzero chemical potential and random matrix theory*, Phys. Rev. Lett., 97 (2006), Art. 012003, 4 pages.
[14] J. P. BOYD, *Chebyshev and Fourier Spectral Methods*, 2nd ed., Dover, Mineola, 2001.
[15] J. BRANNICK, A. FROMMER, K. KAHL, B. LEDER, M. ROTTMANN, AND A. STREBEL, *Multigrid preconditioning for the overlap operator in lattice QCD*, Numer. Math., 132 (2016), pp. 463–490.
[16] C. W. CLENSHAW, *A note on the summation of Chebyshev series*, Math. Tables Aids Comput., 9 (1955), pp. 118–120.
[17] A. CORTINOVIS, D. KRESSNER, AND Y. NAKATSUKASA, *Speeding Up Krylov subspace methods for computing $f(A)b$ via randomization*, SIAM J. Matrix Anal. Appl., 45 (2024), pp. 619–633.
[18] T. A. DAVIS AND Y. HU, *The University of Florida sparse matrix collection*, ACM Trans. Math. Software, 38 (2011), Art. 1, 25 pages.
[19] E. DEADMAN, N. J. HIGHAM, AND R. RALHA, *Blocked Schur algorithms for computing the matrix square root*, in International Workshop on Applied Parallel Computing, P. Manninen and P. Öster, eds., Lect. Notes Comput. Sci., vol 778, Springer, Berlin, 2012, pp. 171–182.
[20] V. DRUSKIN AND L. KNIZHNERMAN, *Gaussian spectral rules for the three-point second differences. I. A two-point positive definite problem in a semi-infinite domain*, SIAM J. Numer. Anal., 37 (2000), pp. 403–422.
[21] P. F. DUBOIS, A. GREENBAUM, AND G. H. RODRIGUE, *Approximating the inverse of a matrix for use in iterative algorithms on vector processors*, Computing, 22 (1979), pp. 257–268.
[22] M. EIERMANN AND O. G. ERNST, *A restarted Krylov subspace method for the evaluation of matrix functions*, SIAM J. Numer. Anal., 44 (2006), pp. 2481–2504.
[23] M. EIERMANN, O. G. ERNST, AND S. GÜTTEL, *Deflated restarting for matrix functions*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 621–641.
[24] M. EMBREE, J. A. LOE, AND R. MORGAN, *Polynomial preconditioned Arnoldi with stability control*, SIAM J. Sci. Comput., 43 (2021), pp. A1–A25.
[25] E. ESTRADA, *Path Laplacians versus fractional Laplacians as nonlocal operators on networks*, New J. Phys., 23 (2021), Art. 073049, 20 pages.
[26] L. FOX AND I. B. PARKER, *Chebyshev Polynomials in Numerical Analysis*, Oxford University Press, London, 1968.
[27] R. FREUND, *On conjugate gradient type methods and polynomial preconditioners for a class of complex non-Hermitian matrices*, Numer. Math., 57 (1990), pp. 285–312.
[28] A. FROMMER, S. GÜTTEL, AND M. SCHWEITZER, *Convergence of restarted Krylov subspace methods for Stieltjes functions of matrices*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 1602–1624.
[29] ———, *Efficient and stable Arnoldi restarts for matrix functions based on quadrature*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 661–683.
[30] A. FROMMER, K. KAHL, M. SCHWEITZER, AND M. TSOLAKIS, *Krylov subspace restarting for matrix Laplace transforms*, SIAM J. Matrix Anal. Appl., 44 (2023), pp. 693–717.
[31] A. FROMMER AND M. SCHWEITZER, *Error bounds and estimates for Krylov subspace approximations of Stieltjes matrix functions*, BIT Numer. Math, 56 (2016), pp. 865–892.
[32] A. FROMMER AND V. SIMONCINI, *Matrix functions*, in Model Order Reduction: Theory, Research Aspects and Applications, W. H. A. Schilders, H. A. van der Vorst, and J. Rommes, eds., Math. Ind., 13, Eur. Consort. Math. Ind. (Berl.), Springer, Berlin, 2008, pp. 275–303.
[33] C. GATTRINGER AND C. LANG, *Quantum Chromodynamics on the Lattice: An Introductory Presentation*, Springer, Berlin, 2010.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

[34] S. GÜTTEL, *Rational Krylov Methods for Operator Functions*, PhD. Thesis, Technische Universität Bergakademie Freiberg, Freiberg, 2010.
[35] S. GÜTTEL, D. KRESSNER, AND K. LUND, *Limited-memory polynomial methods for large-scale matrix functions*, GAMM-Mitt., 43 (2020), Art. e202000019, 19 pages.
[36] S. GÜTTEL AND M. SCHWEITZER, *A comparison of limited-memory Krylov methods for Stieltjes functions of Hermitian matrices*, SIAM J. Matrix Anal. Appl., 42 (2021), pp. 83–107.
[37] ———, *Randomized sketching for Krylov approximations of large-scale matrix functions*, SIAM J. Matrix Anal. Appl., 44 (2023), pp. 1073–1095.
[38] P. HENRICI, *Applied and Computational Complex Analysis. Vol. 3*, Wiley, New York, 1986.
[39] P. HERNÁNDEZ, K. JANSEN, AND M. LÜSCHER, *Locality properties of Neuberger's lattice Dirac operator*, Nuclear Phys. B, 552 (1999), pp. 363–378.
[40] N. J. HIGHAM, *Functions of Matrices*, SIAM, Philadelphia, 2008.
[41] R. A. HORN AND C. R. JOHNSON, *Topics in Matrix Analysis*, Cambridge University Press, Cambridge, 1991.
[42] M. ILIĆ, I. W. TURNER, AND D. P. SIMPSON, *A restarted Lanczos approximation to functions of a symmetric matrix*, IMA J. Numer. Anal., 30 (2010), pp. 1044–1061.
[43] W. JOUBERT, *A robust GMRES-based adaptive polynomial preconditioning algorithm for nonsymmetric linear systems*, SIAM J. Sci. Comput., 15 (1994), pp. 427–439.
[44] JÜLICH SUPERCOMPUTING CENTRE, *JUWELS—-Jülich Wizard for European Leadership Science*. https://www.fz-juelich.de/en/ias/jsc/systems/supercomputers/juwels
[45] C. LANCZOS, *Chebyshev polynomials in the solution of large-scale linear systems*, in Proceedings of the Association for Computing Machinery, Toronto, 1952, Sauls Lithograph Co., Washington, 1953, pp. 124–133.
[46] Q. LIU, R. B. MORGAN, AND W. WILCOX, *Polynomial preconditioned GMRES and GMRES-DR*, SIAM J. Sci. Comput., 37 (2015), pp. S407–S428.
[47] J. A. LOE, *Polynomial Preconditioning with the Minimum Residual Polynomial*, PhD. Thesis, Baylor University, Waco, 2020.
[48] J. A. LOE AND R. B. MORGAN, *Toward efficient polynomial preconditioning for GMRES*, Numer. Linear Algebra Appl., 29 (2022), Art. e2427, 21 pages.
[49] J. A. LOE, H. K. THORNQUIST, AND E. G. BOMAN, *Polynomial preconditioned GMRES in Trilinos: practical considerations for high-performance computing*, in Proceedings of the 2020 SIAM Conference on Parallel Processing for Scientific Computing, G. Biros and U. Meier Yang, eds., SIAM, Philadelphia, 2020, pp. 35–45.
[50] R. METZLER AND J. KLAFTER, *The random walk's guide to anomalous diffusion: a fractional dynamics approach*, Phys. Rep., 339 (2000), pp. 1–77.
[51] I. MORET AND P. NOVATI, *RD-rational approximations of the matrix exponential*, BIT Numer. Math., 44 (2004), pp. 595–615.
[52] N. M. NACHTIGAL, L. REICHEL, AND L. N. TREFETHEN, *A hybrid GMRES algorithm for nonsymmetric linear systems*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 796–825.
[53] Y. NAKATSUKASA AND J. A. TROPP, *Fast & accurate randomized algorithms for linear systems and eigenvalue problems*, SIAM J. Matrix Anal. Appl., 45 (2024), pp. 1183–1214.
[54] H. NEUBERGER, *Exactly massless quarks on the lattice*, Phys. Lett. B, 417 (1998), pp. 141–144.
[55] D. PALITTA, M. SCHWEITZER, AND V. SIMONCINI, *Sketched and truncated polynomial Krylov methods: evaluation of matrix functions*, Preprint on arXiv, 2023. https://arxiv.org/abs/2306.06481
[56] A. N. PETTITT, I. S. WEIR, AND A. G. HART, *A conditional autoregressive Gaussian process for irregularly spaced multivariate data with application to modelling large sets of binary data*, Stat. Comput., 12 (2002), pp. 353–367.
[57] G. PLEISS, M. JANKOWIAK, D. ERIKSSON, A. DAMLE, AND J. GARDNER, *Fast matrix square roots with applications to Gaussian processes and Bayesian optimization*, in Advances in Neural Information Processing Systems 33, H. Larochelle, M. A. Ranzato, R. Hadsell, M.-F. Balcan, and H.-T. Lin, eds., NeurIPS, 2020, pp. 22268–22281.
[58] J. E. ROMAN, C. CAMPOS, L. DALCIN, E. ROMERO, AND A. TOMÁS, *SLEPc users manual*, Tech. Rep. DSIC-II/24/02, Universitat Polytècnica di València, València, 2023.
[59] W. RUDIN, *Real and Complex Analysis*, 3rd ed., McGraw-Hill, New York, 1987.
[60] Y. SAAD, *Practical use of polynomial preconditionings for the conjugate gradient method*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 865–881.
[61] ———, *Least squares polynomials in the complex plane and their use for solving nonsymmetric linear systems*, SIAM J. Numer. Anal., 24 (1987), pp. 155–169.
[62] ———, *Numerical Methods for Large Eigenvalue Problems*, rev. ed., SIAM, Philadelphia, 2011.
[63] G. L. G. SLEIJPEN AND H. A. VAN DER VORST, *A Jacobi-Davidson iteration method for linear eigenvalue problems*, SIAM Rev., 42 (2000), pp. 267–293.
[64] H. TAL-EZER, *On restart and error estimation for Krylov approximation of* $w = f(A)v$, SIAM J. Sci. Comput., 29 (2007), pp. 2426–2441.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

[65] H. K. THORNQUIST, *Fixed-Polynomial Approximate Spectral Transformations for Preconditioning the Eigenvalue Problem*, PhD. Thesis, Rice University, Houston, 2006

[66] E. TIMSIT, L. GRIGORI, AND O. BALABANOV, *Randomized orthogonal projection methods for Krylov subspace solvers*, Preprint on arXiv, 2023. https://arxiv.org/abs/2302.07466

[67] J. VAN DEN ESHOF, A. FROMMER, TH. LIPPERT, K. SCHILLING, AND H. A. VAN DER VORST, *Numerical methods for the QCD overlap operator. I. Sign-function and error bounds*, Comput. Phys. Commun., 146 (2002), pp. 203–224.

[68] J. VAN DEN ESHOF AND M. HOCHBRUCK, *Preconditioning Lanczos approximations to the matrix exponential*, SIAM J. Sci. Comput., 27 (2006), pp. 1438–1457.

[69] D. P. WOODRUFF, *Sketching as a tool for numerical linear algebra*, Found. Trends Theor. Comput. Sci., 10 (2014), pp.1–157.

[70] X. YE, Y. XI, AND Y. SAAD, *Proxy-GMRES: preconditioning via GMRES in polynomial space*, SIAM J. Matrix Anal. Appl., 42 (2021), pp. 1248–1267.