BERGISCHE
UNIVERSITÄT
WUPPERTAL

Department of Mathematics,
and Natural Sciences

Master-Thesis

Accelerating the computation of the matrix sign function

Jay Karippacheril Jacob
2130800

Computer Simulation in Science
Computational Fluid Mechanics

Wuppertal, 03. December 2024

If you fail, never give up because FAIL means "First Attempt in Learning."

(APJ Abdul Kalam)

Declaration of Authorship

I, Jay Karippacheril Jacob, declare that this thesis titled, "Accelerating the computation of the matrix sign function" and the work presented in it are my own. I confirm that:

- This work was been done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Wuppertal, 03. December 2024

(Signature)

Acknowledgment

First and foremost, I would like to express my heartfelt gratitude to Prof. Dr. Andreas Frommer. From the very first email inquiring about the possibility of working under his supervision to every step throughout the course of this thesis, he has been consistently kind, generous, and supportive. I deeply appreciate his guidance in helping me grasp the critical concepts required for my research and his invaluable assistance in rectifying the mistakes I made along the way. Our weekly meetings were always constructive, enlightening, and filled with engaging discussions, which I truly cherished. It has been a remarkable experience to complete my thesis under his supervision, and I am sincerely grateful for the initial ideas and references he provided that shaped the foundation of this work.

I am also deeply grateful to Dr. Gustavo Alonso Ramirez Hidalgo for his exceptional support and guidance. At the early stages of my thesis, I found certain technical concepts challenging to understand, but his clear explanations helped me navigate through these difficulties. I greatly appreciate the time and effort he invested in reviewing my algorithms and code, ensuring better implementations. Despite leaving the University of Wuppertal, he continued to offer his support, for which I will always be indebted.

I would also like to extend my gratitude to Jose Jimenez Merchan for stepping in during the final stages of my thesis and offering guidance to me.

To Prof. Dr. Andreas Frommer, Dr. Gustavo Alonso Ramirez Hidalgo, and José Jiménez Merchán, I am immensely thankful for their unique personalities, unwavering support, and the time they dedicated to me throughout my master's thesis journey.

Lastly, I would like to express my heartfelt appreciation to my family, friends, and the Faculty of Mathematics and Natural Sciences at the University of Wuppertal. Their constant encouragement, support, and belief in me have been a source of strength throughout my master's journey.

Thank you all for being an integral part of this significant milestone in my life.

Abstract

The matrix sign function arises in computations in lattice QCD. We look at the computation of the action $\text{sign}(Q)x$ of the sign function of the matrix Q on a vector x. In our application, Q is the symmetrized Wilson-Dirac operator. This is a Hermitian matrix if the chemical potential is 0; otherwise, it is non-Hermitian. Actually, we will always consider the inverse square root function, since $\text{sign}(Q)x = (Q^2)^{-1/2}Qx$.

The Arnoldi Krylov subspace approximation is the basis method to approximate $\text{sign}(Q)x$. There are several ways to accelerate the convergence of this basic scheme:

1. **Restarts** (in the non-Hermitian case). This avoids having too many inner products in the Arnoldi orthogonalization.

2. **Deflation** (explicit and implicit). This makes the matrix better conditioned and thus reduces the number of iterations. Explicit deflation uses the smallest left and right eigenvectors; implicit deflation is present in the thick restart approach of Eiermann and Güttel; see also the `funm` Matlab code.

3. **Polynomial preconditioning.** This also makes the matrix better conditioned and thus reduces the number of iterations. A recent paper on this was published along with the numerical results for QCD on a parallel machine.

4. **Sketching.** This is a randomized approach where we save orthogonalizations and sketch the Arnoldi matrix. The relevant paper is by Güttel and Schweitzer.

The purpose of the thesis is to consider the following combination of the above approaches:

- $2 + 1$ (as is already done in `funm`)
- $2 + 3$ (building on existing work and code of Gustavo)
- $2 + 4$ (this is new, but Stefan Güttel just gave a talk on it at a conference in Paris)

Tasks:

1. Understand and describe the individual methods (1–4).

2. Describe, formulate algorithmically, and discuss the combined methods ($2 + 1$, $2 + 3$, $2 + 4$).

3. Test the combined methods, in Matlab on small configurations.

Contents

1 Introduction

Consider a matrix $\mathbf{A} \in \mathbb{C}^{n\times n}$, a vector $\mathbf{b} \in \mathbb{C}^n$ and a function $\mathbf{f} : \mathbb{C} \to \mathbb{C}$, the action of a matrix is defined as:

$$f(A)b \tag{1.1}$$

The above expression represents a product of the matrix function $f(A) \in \mathbb{C}^{n\times n}$ on a vector $\mathbf{b}$. There exists a huge interest in the action of a matrix on a vector in the fields of science and engineering. Some of the most interesting cases widely under studies are :

1. **Matrix exponential function** $f(z) = e^z$, forms the core of exponential integrators used for solving differential equations [HL97; HO10; MVL03].
2. **Matrix square root** $f(z) = z^{1/2}$, in machine learning [PJE+20] and in other domains such as image processing, advection-diffusion problems, elasticity and many more [AL09; ITS09].
3. **Matrix logarithm** $f(z) = \log(z)$, used in Markov model analysis [SS76].
4. **Matrix fractional powers** $f(z) = z^{\alpha}$, in fractional differential equations [BHK12].
5. **Matrix sign function** $f(z) = \mathrm{sign}(z)$, in lattice quantum chromodynamics (QCD) [EFL+02; BFLW07].

The most straightforward approach to compute $f(A)\mathbf{b}$ is to first calculate $f(A)$ and then perform matrix multiplication with $\mathbf{b}$. However, as the dimension of the matrix grows, this approach becomes impractical due to various reasons such as the storage complexity, computational cost of matrix functions, and inefficiency of matrix-vector multiplication.

Here, our domain of interest is the Matrix sign function, in conjunction with the application of lattice QCD. The most significant challenge faced in lattice QCD was the implementation of chiral symmetry on the lattice [FX23] and one among the prominent solutions proposed to overcome this was the Overlap-Dirac operator involving the sign function, which avoids low mode calculation for chiral symmetry [NT22]. However, the drawback of the above proposal was the huge computational cost of the matrix sign function since the matrix A is a large sparse matrix. Typically the matrix A is Hermitian and efficient methods have been already developed to approximate them as mentioned in papers [Neu98; EFL+02].

Studying the relativistic heavy ion collisions theoretically in lattice simulations and model calculations implies presenting a non-zero density. As a result, a quark chemical potential is introduced to the QCD Lagrangian, leading to the loss of Hermiticity of the matrix A as in [BW06a]. Therefore, we are now faced with the computation of the explained matrix sign function for a non-Hermitian matrix A.

For smaller lattices, the existing methods could be used in the above-mentioned problem. However, as the dimension of the matrix becomes larger, one has to heavily depend upon the iterative methods for approximating the matrix sign function. Some of the popular methods under use in such situations are polynomial [DK89; Saa92] and rational [DK98; Güt13;

GK13] Krylov methods which demand, a high arithmetic cost for the orthogonalization of a Krylov basis or the large memory cost for the storage of Krylov basis vectors. These limitations narrow down the attainable accuracy of the Krylov methods. To address these constraints, there are several strategies available. Among them a popular strategy is the utilization of other subspaces with superior approximation properties.

In this paper, we explore new possibilities arising from the combination of deflation and Krylov subspace methods, chosen for their strengths in relation to the matrix sign function and specific applications of interest. In Chapter 2, we begin with an introduction to matrix functions, including essential definitions and properties. This discussion narrows in Chapter 3, where we focus on the matrix sign function, our primary area of investigation. Here, we cover definitions and properties derived from matrix functions, along with specific characteristics unique to the matrix sign function.

Chapter 4 provides a concise overview of Quantum Chromodynamics (QCD) simulations, particularly the Wilson-Dirac and overlap operators in lattice QCD. We examine the limitations encountered in calculating sign functions in this context, highlighting the challenges and the motivation to develop algorithms that enhance efficiency and stability.

Our goal is to approximate the action of a matrix sign function on a vector more efficiently and stably. To this end, we introduce recent methods identified in our literature review, alongside algorithms for their implementation, in Chapters 5 and 6. In Chapter 7, we present a framework for implementing potential new algorithms and discuss the rationale behind the choices and combinations selected for our numerical experiments. Finally, Chapter **??** provides an in-depth analysis of the performance of these algorithms across various parameters.

2 Matrix Functions

As a starting point for this thesis, we begin our research by reviewing the existing literature on our domain of interest, the action of a matrix $f(A)\mathbf{b}$. Specifically, we are focused on algorithms that can expedite the matrix sign functions, $sgn(A)$ for large non-Hermitian matrices with dimension N.To achieve this objective, we begin by presenting fundamental information on matrix functions. This is followed by a discussion of the matrix sign function in Chapter 3.

In the prior Chapter, we indicated our interest in large non-Hermitian matrices of dimension N. Hence, with the help of chapter 4, we will elucidate the reason behind the focus on this particular class of matrices. Proceeding further in this review, we aim to address algorithms that could potentially minimize the two significant computational challenges: cost and time. Chapter 5 introduces Krylov methods, including the Arnoldi method, the restarted Arnoldi method, and the sketched Arnoldi method, which are of particular interest for further research. Chapter 6 will explore deflation along with the LR-deflation algorithm, another area of keen interest.

Throughout this Chapter, we will anchor our discussion on [Hig08], which provides a robust foundation for the theory of matrix functions. As outlined in this reference, although there exist different ways of defining $f(A)$ there are three definitions we are interested in the context of computing $f(A)$.

2.1 Definitions of $f(A)$

Definition 2.1.1. *[Hig08](Jordan canonical form). Any matrix $A \in \mathbb{C}^{n\times n}$ can be written in the Jordan canonical form,*

$$Z^{-1}AZ = J = diag(J_1, J_2, \dots, J_p), \tag{2.1}$$

$$J_k = J_k(\lambda_k) = \begin{bmatrix} \lambda_k & 1 & & \\ & \lambda_k & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_k \end{bmatrix} \in \mathbb{C}^{m_k\times m_k}. \tag{2.2}$$

where Z is non-singular and $m_1 + m_2 + ... + m_p = n$.

In the above standard result, the Jordan matrix J is unique up to the ordering of the blocks J_i, whereas Z known as the transforming matrix is not unique. Here, $\lambda_1, ..., \lambda_p$ denotes the distinct eigenvalues of the matrix A used to formulate Jordan blocks, with n_i representing the size of the largest Jordan block containing eigenvalue λ_i.

Before presenting the definition of matrix functions via Jordan canonical form, we first introduce the following terminology.

Definition 2.1.2. *[Hig08]The function f is said to be defined on the spectrum of A if the values*

$$f^{(j)}(\lambda_i), \quad j = 0 : n_i - 1, \quad i = 1 : s$$

exist. These are called the values of the function f on the spectrum of A.

The following definition of matrix functions via the Jordan canonical form depends solely on the values of f evaluated at the spectrum of A, without requiring additional information beyond this spectrum. Indeed, any $\sum_{i=1}^{s} n_i$ arbitrary values can be chosen and assigned as the values of f on the spectrum of A. Only when making statements about global properties, such as continuity, do we need to impose additional assumptions on f.

Definition 2.1.3. *[Hig08](matrix function via Jordan canonical form). Let f be defined on the spectrum of $A \in \mathbb{C}^{n\times n}$ and let A have the Jordan canonical form 2.1 and 2.2. Then*

$$f(A) := Zf(J)Z^{-1} = Z\text{diag}(f(J_k))Z^{-1} \tag{2.3}$$

where

$$f(J_k) := \begin{bmatrix} f(\lambda_k) & f'(\lambda_k) & \cdots & \frac{f^{(m_k-1)}(\lambda_k)}{(m_k-1)!} \\ & f(\lambda_k) & \ddots & \vdots \\ & & \ddots & f'(\lambda_k) \\ & & & f(\lambda_k) \end{bmatrix} \tag{2.4}$$

.

The insights we infer from the first definition for $f(A)$ are:

1. $f(A)$ is independent of the Jordan canonical form used.

2. If A is diagonalizable then the Jordan canonical form reduces to an eigendecomposition $A = ZDZ^{-1}$, with $D = diag(\lambda_i)$ and the columns of Z are eigenvectors of A

The Jordan canonical form is rarely used in computations due to its high sensitivity to perturbations. However, in the special case where A is normal (i.e., unitarily diagonalizable), the second inference from the aforementioned definition becomes applicable and $f(A)$ could be computed from the well-conditioned eigendecomposition. This direct method of computing $f(A)$ is therefore employed only when A is a small Hermitian matrix, with a computational complexity of $O(n^3)$ [Hig08].

The second approach for defining $f(A)$ is with the help of polynomial interpolation, which yields numerous useful properties.

Theorem 2.1.4. *[Hig08]For polynomials p and q and $A \in \mathbb{C}^{n\times n}$, $p(A) = q(A)$ if and only if p and q take the same values on the spectrum of A.*

The above theorem establishes that the matrix $p(A)$ is entirely determined by the values of p on the spectrum of A.

Definition 2.1.5. *[Hig08](matrix function via Hermitian interpolation). Let f be defined on the spectrum of $A \in \mathbb{C}^{n\times n}$ and let ψ be the minimal polynomial of A, where $\psi(x) = \prod_{i=1}^{s}(x-\lambda_i)^{n_i}$ Then $f(A) := p(A)$, where p is the polynomial of degree less than*

$$\sum_{i=1}^{s} n_i = \deg \psi$$

that satisfies the interpolation conditions

$$p^{(j)}(\lambda_i) = f^{(j)}(\lambda_i), \quad j = 0 : n_i - 1, \quad i = 1 : s \tag{2.5}$$

There is a unique such p with minimal degree and it is known as the Hermite interpolation polynomial.

While the second definition of $f(A)$ appears more numerically practical than the first, it is important to note that the interpolating polynomial p not only depends on f but also on the eigenvalues of A. As cited in [Tso23], it would necessitate $O(n^4)$ floating point operations ($(O(n)$ matrix-matrix multiplications each of which costs $O(n^3)$) to produce $f(A)$ and is numerically unstable.

Remark 2.1.6. *Some important remarks on the above definition based on [Hig08]are:*

1. *If the polynomial q satisfies the interpolation conditions specified in Equation 2.5 as well as additional interpolation conditions (whether at the same or different λ_i), then q and the polynomial p from Definition 2.1.5 yield identical values on the spectrum of A. Consequently, by Theorem 2.1.4, it follows that $q(A) = p(A) = f(A)$.*

2. *The Hermite interpolating polynomial p can be defined explicitly by the Lagrange–Hermite formula*

$$p(t) = \sum_{i=1}^{s}\left[\left(\sum_{j=0}^{n_i-1}\frac{1}{j!}\phi_i^{(j)}(\lambda_i)(t-\lambda_i)^j\right)\prod_{\substack{j=1\\ j\neq i}}^{s}(t-\lambda_j)^{n_j}\right] \tag{2.6}$$

   where $\phi_i(t) = \frac{f(t)}{\prod_{j\neq i}(t-\lambda_j)^{n_j}}$.

3. *The definition explicitly makes $f(A)$ a polynomial in A.*

4. *According to Definition 2.1.5, even if f is represented by a power series, $f(A)$ can still be expressed as a polynomial in A of degree at most $n-1$.*

5. *If A is a real, diagonal matrix, then for the condition $f(A)$ to be real whenever A is real becomes evident only when the scalar function f is real on the subset of the real line on which it is defined.*

6. *The Definition 2.1.5 could be directly derived from the formula mentioned in equation 2.4 for a function of the Jordan block* J_k. *We can directly derive from Definition 2.1.5 the formula 2.4 for a function of the Jordan block* J_k. *The sufficient interpolation conditions to achieve the Hermite interpolating polynomial,*

$$p(t) = f(\lambda_k) + f'(\lambda_k)(t - \lambda_k) + \frac{f''(\lambda_k)}{2!}(t - \lambda_k)^2 + \cdots + \frac{f^{(m_k-1)}(\lambda_k)}{(m_k - 1)!}(t - \lambda_k)^{m_k-1}.$$

The third approach of defining $f(A)$ involves the Cauchy integral theorem, assuming f is analytic, unlike the other two definitions where f has to be defined on the spectrum of A.

Definition 2.1.7. *[Hig08](matrix function via Cauchy integral). For* $A \in \mathbb{C}^{n\times n}$,

$$f(A) := \frac{1}{2\pi i}\int_\Gamma f(z)(zI - A)^{-1}dz \tag{2.7}$$

where f *is analytical on and inside a closed contour* Γ *that encloses* $spec(A)$.

The above definition is highly applicable to our problem of interest. Here we face many numerical challenges and the most critical challenge encountered is the identification of an appropriate contour Γ and a quadrature rule that depends on both f and A.

Thus, a good definition is one that can be chosen to not only yield the expected properties but also reveal useful, less obvious ones. Accordingly, we conclude this chapter by presenting some general properties derived from the definition of $f(A)$.

Remark 2.1.8. *(properties of the matrix functions)[Hig08]*

1. $f(A)$ *commutes with* A.
2. $f(A^T) = f(A)^T$.
3. $f(XAX^{-1}) = Xf(A)X^{-1}$.
4. *The eigenvalues of* $f(A)$ *are* $f(\lambda_i)$, *where the* λ_i *are the eigenvalues of* A.
5. *If* X *commutes with* A *then* X *commutes with* $f(A)$.

3 Matrix Sign Function

To introduce the matrix sign function, it is essential to explore the scalar sign function as it represents an extension of their scalar counterparts. The scalar sign function is defined over the complex plane excluding the imaginary axis: $\mathbb{C} \setminus \mathbb{C}^0 = \mathbb{C}^+ \cup \mathbb{C}^-$ where $\mathbb{C}^-$, $\mathbb{C}^+$ and $\mathbb{C}^0$ denote the open right-half complex plane, the open left-half complex plane, and the imaginary axis, respectively. Thus, the scalar sign function for $z \in \mathbb{C}^+ \cup \mathbb{C}^-$ is defined by[KL95]

$$\operatorname{sign} z = \begin{cases} 1, & z \in \mathbb{C}^+ \\ -1, & z \in \mathbb{C}^- \end{cases} \tag{3.1}$$

The above definition implies that if $z \in \mathbb{C}^0 = 0$ then $\operatorname{sign}(z)$ is undefined. Now based on the above, to define the matrix sign function, we proceed under the assumption that $A \in \mathbb{C}^{n\times n}$ does not possess eigenvalues lying on the imaginary axis, thereby ensuring A is non-singular and $\operatorname{sign}(A)$ remains well-defined. These conditions are crucial for establishing the validity of $\operatorname{sign}(A)$. There exist numerous equivalents for matrix sign functions that extend meaningful insights into the properties they possess. In the following subsection, we will examine several pertinent definitions essential to this thesis.

3.1 Definition of $\operatorname{sign}(A)$

Definition 3.1.1. *[Rob80](Jordan Canonical Form) Let the matrix A have a Jordan decomposition*

$$A = T \begin{bmatrix} N & 0 \\ 0 & P \end{bmatrix} T^{-1}$$

where N and P are square matrices with eigenvalues in $\mathbb{C}^-$ and $\mathbb{C}^+$, respectively. Then the sign of A is defined to be,

$$\operatorname{sign}(A) = T \begin{bmatrix} -I_N & 0 \\ 0 & I_P \end{bmatrix} T^{-1} \tag{3.2}$$

where the identity matrices I_N, and I_P, are compatibly dimensioned with N and P, respectively.

The above definition is a derivation of Definition 2.1.3, where the matrix function is the sign function, and for this function, all derivatives of all orders are zero.

Some intriguing properties for $\operatorname{sign}(A)$ derived from the above initial definition are highlighted in the following remark.

Remark 3.1.2. *(properties of the sign function)[KL95; Rob80]*

1. $\operatorname{sign}(A)$ *is diagonalizable with eigenvalues equal to* ± 1.

2. $\operatorname{sign}(A)^2 = I$.
3. *If c is a nonzero real scalar, then* $\operatorname{sign}(cA) = \operatorname{sign}(c)\operatorname{sign}(A)$.
4. $\operatorname{neg}(A) \equiv (I - \operatorname{sign}(A))/2$ *is a projection onto the negative invariant subspace of A and* $\operatorname{pos}(A) \equiv (I + \operatorname{sign}(A))/2$ *is a projection onto the positive invariant subspace of A, where the positive and negative invariant subspaces of A are the subspaces corresponding to the eigenvalues of A in $\mathbb{C}^-$ and $\mathbb{C}^+$ respectively.*

Lemma 3.1.3. *[KL95]Given,*

$$A = U \begin{bmatrix} N & T \\ 0 & P \end{bmatrix} U^T$$

where U is an orthogonal matrix, N has eigenvalues in $\mathbb{C}^-$, and P has eigenvalues in $\mathbb{C}^+$. Then the sign of A is given by,

$$sgn(A) = U \begin{bmatrix} -I_N & S \\ 0 & I_P \end{bmatrix} U^T \tag{3.3}$$

where S satisfies the Sylvester equation,

$$NS - SP = -2T \tag{3.4}$$

.

The above formulation proves to be highly beneficial. It can be used to analyze the stability of Newton iteration [Bye86; BHM97] and analyze the conditioning [BHM97] of the matrix sign function. This definition further serves as a foundation for a method of solving the stable Sylvester equation of the form 3.4. In the equation 3.3 replace U with I to determine $\operatorname{sign}(A)$. Now, the upper right block which is S of $\operatorname{sign}(A)$ is the solution desired to be found from the above-stated Sylvester equation.

The second type of definition is based on integral representations. Utilizing a residue argument, presented in the spectral theory of operators, Robert derives an integral formula of the form [Rob80],

$$\operatorname{pos}(A) = \frac{1}{2\pi i} \int_D (\zeta I - A)^{-1} d\zeta \tag{3.5}$$

where D is a simple closed contour in $\mathbb{C}^+$ containing the eigenvalues of A with positive real part and pos(A) as mentioned in remark 3.1.2. From this equation and the remark 3.1.2 Robert derived an integral representation as a definition for $\operatorname{sign}(A)$,

Definition 3.1.4.

$$\operatorname{sign}(z) = \frac{2}{\pi} z \int_0^{+\infty} (y^2 I + z^2)^{-1} dy \tag{3.6}$$

Definition 2.1.7 and the above definition are identical when $f(z) \equiv 1$ for the contour $\mathcal{C}^+$ in Definition 2.1.7.

The third method of defining sign(A) is through matrix iterations. Newton's iteration is the most popular iterative method to find sign(A). The method is applied to the equation $S^2 - I = 0$. Let $A_0 = A$ and set,

$$A_{k+1} = \frac{1}{2}(A_k + A_k^{-1}) \tag{3.7}$$

The above matrix iteration is globally convergent for all matrices A with eigenvalues in $\mathbb{C}^- \cup \mathbb{C}^+$.

$$sgn(A) = \lim_{k \longrightarrow +\infty} A_k \tag{3.8}$$

An intriguing aspect of Newton's iterative method is that the convergence is quadratic when A_k is close to the actual A but could be relatively slow at the initial stages.

Higher-order Padé iterative methods are another form of iterative methods used for the evaluation of sign(A). The general form of the equation used in these iterations for order n is,

$$A_{k+l} = P_n(A_k)Q_n^{-1}(A_k) \tag{3.9}$$

where $P_n(A)$ and $Q_n(A)$ are the odd and even parts respectively of the polynomial $(I + A)^n$.The Padé iterations are globally convergent and serve as an implicit definition for sign(A). Introduction of the tanh identity in equation 3.9 helps in the study of chaotic behaviours of sign(A) on the eigenvalues of A close to the imaginary axis[KL94].

$$P_n(A_k)Q_n^{-1}(A_k) = \tanh(n \operatorname{arctanh}(A_k)) \tag{3.10}$$

If we represent x in polar form, i.e., $x = re^{i\phi}$, then x has two principal branches, given by $\sqrt{x} = \sqrt{r}e^{i\frac{\phi}{2}}$ for $\phi \in [-\pi, \pi]$. Following the first type of definition and extending the scalar formulation of the sign function, we define sign$(z) = \frac{z}{\sqrt{z^2}}$ as presented by Higham [Hig94] and apply it to the corresponding matrix function. This extension holds only when z is non-zero and consequently, when extended to a matrix A, the matrix must have no purely imaginary eigenvalues.

For such matrices, A^2 contains no eigenvalues on the negative real axis, thus ensuring that there exists a unique square root, $N = (A^2)^{\frac{1}{2}}$. This commutes with A and has eigenvalues in the open right-half complex plane as cited in the paper[DJ74]. Thus we have a definition for the matrix sign function:

Definition 3.1.5. *Let the spec$(A) \cap \mathbb{R}^- = \emptyset$, then*

$$\operatorname{sign}(A) = A(A^2)^{-\frac{1}{2}} \tag{3.11}$$

4 QCD simulations and its Non-Hermitian challenges

One of the most demanding applications for supercomputers currently is Lattice QCD simulation, where a significant amount of resources are allocated. Quantum chromodynamics (QCD) is a quantum field theory for the strong interaction of the quarks via gluons[KMA+22]. This theory is applied to make predictions on masses and resonance spectra on hadrons[DFF+08].

4.1 The Wilson-Dirac and the overlap operator in lattice QCD

The governing equation that determines the dynamics of the quarks and the interaction of quarks and gluons is the Dirac equation.

$$D\psi + m \cdot \psi = \eta \tag{4.1}$$

In the above equation the quark fields are represented by $\psi = \psi(x)$ and $\eta = \eta(x)$, where x denotes the points in space-time, $x = (x_0, x_1, x_2, x_3)$[MM94]. The Dirac operator D in the equation 4.1 represents the gluons and sets the mass of the quarks in the QCD theory. The parameter m is a scalar mass. The Dirac operator can be written as:

$$D = \sum_{\mu=0}^{3} \gamma_\mu \otimes (\partial_\mu + A_\mu) \tag{4.2}$$

where, $\partial_\mu = \partial/\partial x_\mu$ and A is the gluon gauge field with the anti-hermitian traceless matrices $A_\mu(x)$. The γ-matrices represent the generators of the Clifford algebra[BFK+16]. At a given point x, the quark field ψ is expressed by a twelve-component column vector. These column vectors correspond to three colours and four spins, acted upon by $A_\mu(x)$ and γ_μ respectively.

To align with our study, we rewrite the massless overlap Dirac operator with a non-zero chemical potential (μ) as follows[NN81]:

$$D_{ov}(\mu) = 1 + \gamma_5 sgn(H_w(\mu)) \tag{4.3}$$

where,$H_w(\mu) = \gamma_5 D_w(\mu)$, $D_w(\mu)$ is the Wilson-Dirac operator at nonzero chemical potential[HK84; KMS+83] with negative Wilson mass $m_w \in (-2, 0)$, $\gamma_5 = \gamma_1\gamma_2\gamma_3\gamma_4$. The Wilson-Dirac operator is a discretization of the Dirac operator on a four-dimensioned lattice given as,

$$\begin{aligned}[D_w(\mu)]_{nm} = \delta_{n,m} & \\ & - \kappa \sum_{j=1}^{3}(1+\gamma_j)U_{n,j}\delta_{n+\hat{j},m} - \kappa \sum_{j=1}^{3}(1-\gamma_j)U^{\dagger}_{n-\hat{j},j}\delta_{n-\hat{j},m} \\ & - \kappa(1+\gamma_4)e^{\mu}U_{n,4}\delta_{n+\hat{4},m} - \kappa(1-\gamma_4)e^{-\mu}U^{\dagger}_{n-\hat{4},4}\delta_{n-\hat{4},m}\end{aligned} \tag{4.4}$$

where $\kappa = 1/(8+2m_w)$ and $U_{n,v}$ is the $SU(3)$-matrix associated with the link connecting the lattice site n to $n+\hat{v}$. One of the most important highlights of the Wilson-Dirac operator is that compared to the naive discretization of the derivative operator, it avoids the replication of the fermion species for the continuum Dirac operator.

In the discretized formula 4.4 the non-Hermiticity of the operator arises due to the term $e^{\pm\mu}$. The quark field at each lattice site corresponds to 12 variables: 3 $SU(3)$ colour components $\times$ 4 Dirac spinor components. This depicts that the matrix $H_w(\mu)$ inside the sign function shifts its properties from Hermitian to non-Hermitian when $\mu \neq 0$. This means we have a new case to be addressed.

The challenge with non-Hermitian matrices lies in the fact that they typically have complex eigenvalues, which complicates the evaluation of the sign function. Therefore, the application of Definition 3.1.1 to Equation 4.3 necessitates the evaluation of the sign of a complex number. Moreover, Definition 3.1.5 offers a clear understanding of the properties that the sign function must satisfy.

We know that for a square matrix A, $[\text{sign}(A)]^2 = I$ needs to concur for a sign function. A short calculation based on the Jordan block canonical form shows that for the above reason the overlap operator $D_{ov}(\mu)$ as defined in Equation 4.3 satisfies the GinspargWilson relation[BW06b].

$$D_{ov}, \gamma_5 = D_{ov}\gamma_5 D_{ov} \tag{4.5}$$

For A Hermitian, the polar factor $\text{pol}(A) = A(A^{\dagger}A)^{-1/2}$ of A coincides with $\text{sign}(A)$.Building upon the above, significant advancements have been made in developing efficient and faster iterative methods for computing the action of the matrix sign function on a vector. However, for A non-Hermitian, $\text{sign}(A) \neq pol(A)$ and $pol(A)^2 \neq I$. Thus, for $\mu \neq 0$, replacing $\text{sign}(H_w)$ with $\text{pol}(H_w)$ in the definition of the overlap operator in Eq. (1) not only alters the operator but also violates the Ginsparg-Wilson relation, as demonstrated in numerical experiments. We conclude that the definition provided in Equation 4.3 is the correct formulation of the overlap operator for $\mu \neq 0$. This, in turn, generates the motivation for us to explore further iterative methods of sign function of non-Hermitian matrices.

5 Krylov Subspace Methods in Matrix Function Applications

Iterative methods in general play a crucial role in approximating matrix functions efficiently, especially in scenarios where direct calculations are computationally costly and time-consuming. When discussing iterative methods, Krylov subspace methods have garnered significant interest due to extensive research, their properties, and fast convergence. These methods are particularly well-suited for large-scale problems because they produce iterative solutions using only matrix-vector products. In this Chapter, we will introduce some fundamental concepts and algorithms. We will then build upon these basics by exploring selected methods from the Krylov subspace methods that are of particular interest in our research.

To begin, we will first introduce the definition of a Krylov subspace for a matrix A and a vector b, which forms the foundation for everything discussed in the upcoming subsections.

Definition 5.0.1. *[Sch16]The m^{th} Krylov subspace of $A \in \mathbb{C}^{n\times n}$ and $b \in \mathbb{C}^n$ is given by,*

$$\mathcal{K}_m(A, b) := span(b, Ab, A^2b, \dots, A^{m-1}b) = \{p(A)b : p \in \mathcal{P}_{m-1}\},$$

where $\mathcal{P}_{m-1}$ is the set of all polynomials of degree at most $m-1$.

Krylov method works by using the Krylov subspace mentioned in Definition 5.0.1 to find a suitable approximation $f_m \in \mathcal{K}_m(A, b)$ for $f(A)b$. To achieve this, we need to construct a basis for $\mathcal{K}_m(A, b)$. The concept of seeking an approximation to $f(A)b$ within a Krylov subspace $K_m(A, b)$ is naturally motivated by Definition 2.1.2, where every matrix function is essentially a polynomial (of degree at most $n-1$) in A. Therefore, $f(A)b \in K_n(A, b)$. The significant advantage of Krylov subspace methods lies in their inherent capability to obtain good approximations using a polynomial of lower degree, which proves advantageous for our purposes. Some intriguing properties of these methods are highlighted in the following remark.

Remark 5.0.2. *[Sch16]Let $A \in \mathbb{C}^{n\times n}$ and let $b \in \mathbb{C}^n$. In addition, let m^* be the smallest integer such that there exists a polynomial $p_{m^*} \in \Pi_{m^*}$ which satisfies $p_{m^*}(A)b = 0$. Then*

1. $K_m(A, b) \subseteq K_{m+1}(A, b)$ *for all* $m \geq 1$,
2. $K_{m^*}(A, b)$ *is invariant under* A*, and* $K_m(A, b) = K_{m^*}(A, b)$ *for all* $m \geq m^*$,
3. $\dim K_m(A, b) = \min\{m, m^*\}$.

5.1 The Arnoldi approximation for matrix functions

The Arnoldi process is a method within the family of Krylov subspace methods, where the most apparent choice for a basis of $K_m(A, b)$ is the Krylov basis $b, Ab, A^2b, \ldots, A^{m-1}b$. Nevertheless, this basis can exhibit significant ill-conditioning. To ensure numerical stability, we introduce orthogonalization of the basis. Therefore, the method begins by defining $v_1 = \frac{1}{\|b\|_2} b$ and proceeds to construct additional basis vectors through iterative steps, orthogonalizing Av_j against the previous basis vectors $v_1, \ldots, v_{j-1}$. The algorithm developed based on the above idea is as below:

Algorithm 1 Arnoldi process [Saa03]

Require: Matrix $A \in \mathbb{R}^{n \times n}$, vector $b \in \mathbb{R}^n$, integer m
Initialize $v_1 = \frac{1}{\|b\|_2} b$
for $j = 1$ to m **do**
  Compute $w_j = Av_j$
  for $i = 1$ to j **do**
    Compute $h_{i,j} = v_i^H w_j$
    Update $w_j = w_j - h_{i,j} v_i$
  end for
  Compute $h_{j+1,j} = \|w_j\|_2$
  if $h_{j+1,j} = 0$ **then**
    break
  end if
  Set $v_{j+1} = \frac{w_j}{h_{j+1,j}}$
end for
Form matrices $V_m = [v_1, \ldots, v_m]$ and $H_m = [h_{i,j}]_{i,j=1,\ldots,m}$
return V_m, H_m, $h_{m+1,m}$, v_{m+1}

From the aforementioned algorithm, we obtain a matrix $V_m \in \mathbb{C}^{n \times m}$, whose columns consist of the orthonormal basis vectors $v_1, \ldots, v_m$ for $K_m(A, b)$, and an upper Hessenberg matrix $H_m = [h_{i,j}] \in \mathbb{C}^{m \times m}$. These matrices satisfy the Arnoldi relation [Saa03]:

$$AV_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^T \tag{5.1}$$

If $H_m = V_m^H A V_m$, we can infer that for a Hermitian matrix $A = A^H$, the Hessenberg matrix H_m is also Hermitian, and thus tridiagonal. When $h_{i,j} = 0$, the vector v_i is already orthogonal to w_j in Algorithm 1. Simplifying the Arnoldi process according to this observation results in a more cost-effective method known as the Lanczos process, which is considered a special case.

Algorithm 2 Lanczos process [Saa03]

Require: Matrix $A \in \mathbb{R}^{n\times n}$, vector $b \in \mathbb{R}^n$, integer m
Initialize $v_1 = \frac{1}{\|b\|_2} b$
for $j = 1$ to m **do**
  if $j \geq 2$ **then**
    $w_j = Av_j - h_{j,j-1}v_{j-1}$
  else
    $w_j = Av_j$
  end if
  $h_{j,j} = v_j^* w_j$
  $w_j = w_j - h_{j,j}v_j$
  $h_{j+1,j} = \|w_j\|_2$
  if $h_{j+1,j} = 0$ **then**
    break
  end if
  $v_{j+1} = \frac{w_j}{h_{j+1,j}}$
end for
Form matrices $V_m = [v_1, \dots, v_m]$ and $H_m = [h_{i,j}]_{i,j=1,\dots,m}$
return V_m, H_m, $h_{m+1,m}$, v_{m+1}

From the above, we have found a way to construct an orthogonal basis for $\mathcal{K}_m(A, b)$. However, our goal is to approximate $f(A)$ using this method. Hence, we need a procedure to achieve $f(A)b \approx f_m \in \mathcal{K}_m(A, b)$. From Definition 5.0.1 and our prior explanation, we understand that the idea behind any Krylov method is to approximate a polynomial p by a smaller polynomial of degree $m-1$. Thus, we can rephrase our problem of approximating f_m as how to choose a polynomial $p_{m-1} \in \Pi_{m-1}$ such that $p_{m-1}(A)b \approx p(A)b \approx f(A)b$.

We know from the definitions of matrix functions that p interpolates f at $\operatorname{spec}(A)$ and we consider the approximating polynomial p_{m-1} to interpolate f at m suitably chosen points. This leads us to the Ritz values corresponding to $\mathcal{K}_m(A, b)$, which are the eigenvalues of H_m. These eigenvalues are always related to some form of spectral information of A, as they lie within its field of values (which reduces to the spectral interval $[\lambda_{\min}, \lambda_{\max}]$ in the Hermitian case). Moreover, they become exact eigenvalues of A when the Krylov subspace reaches its maximum possible dimension[**39**].

The highlight of choosing p_{m-1} as the polynomial that interpolates f at the Ritz values corresponding to $\mathcal{K}_m(A, b)$ is that $p_{m-1}(A)b$ arise as a by-product without the explicit need to compute p_{m-1}. The above is provided in the below lemma.

Lemma 5.1.1. *[Hig08] Let $A \in \mathbb{C}^{n\times n}$ and let $b \in \mathbb{C}^n$. Let V_m, H_m fulfil the relation 5.1, and let*

$$f_m = V_m f(V_m^H A V_m) V_m^H b = b_2 V_m f(H_m)\hat{e_1} \tag{5.2}$$

where $\hat{e_1}$ is the first unit vector in a coordinate system. Then

$$f_m = p_{m-1}(A)b,$$

where $p_{m-1} \in \Pi_{m-1}$ is the unique polynomial interpolating f at the eigenvalues of H_m in the Hermite sense, provided that f is defined on spec(H_m).

The approximation $f_m = p_{m-1}(A)b$ is considered close to the correct value $f(A)b$ if the m Ritz values are near the n eigenvalues of A. From $H_m = V_m^H A V_m$, it follows that the eigenvalues of H_m lie within the numerical range of A, i.e.,

$$spec(H_m) \subseteq W(A) := \{x^* Ax : \|x\|_2 = 1\}$$

Since spec$(A) \subseteq W(A)$, the Arnoldi approximation 5.2 is a reasonable approach. Furthermore, we observe that the eigenvalues of H_m eventually become eigenvalues of A with an increase in m. In other words, the Arnoldi approximation becomes exact after a finite number of iterations. i.e., in reference to remark 5.0.2, the Arnoldi process is feasible up to m and only then breaks down. We also have

$$\text{spec}(H_m) \subseteq \text{spec}(A), \quad f(A)b = b_2 V_m f(H_m)\hat{e_1}, \tag{5.3}$$

i.e., the Arnoldi approximation is exact for m.

The merits of having the Arnoldi approximation is that we do not need to store A explicitly because we only require A for matrix-vector multiplication at a cost of $\mathcal{O}(n)$. This makes it particularly advantageous for dealing with large sparse matrices.

While we recognize the benefits of Arnoldi approximations, a significant challenge is storing the full matrix V_m, even for large sparse matrices where storing full matrices was previously unnecessary. This challenge applies to sparse and Hermitian or non-Hermitian matrices A, leading to memory limitations after m iterations, depending on the matrix size. Some approaches to mitigate these issues include:

1. For Hermitian matrices and Non-Hermitian matrices, it is well known that the computation for a particular case matrix function can be improved by deflating the eigenvalues smallest in absolute value [EFL+02]. The idea is to treat these critical eigenvalues exactly and perform the Krylov subspace approximation on a deflated space.

2. Recently, [GS23] introduced a method, Randomized Subspace Embedding, that partly avoids orthogonalization in the Arnoldi process by leveraging randomized subspace embedding techniques [MT20]. This approach represents $f(A)$ via a Cauchy integral as defined prior in Definition 2.1.7, thereby reducing the problem of $f(A)b$ to solving shifted inverses $(A + sI)^{-1}b$. Subsequently, Krylov subspace methods for inverses are accelerated through a sketch-and-solve approach akin to [NT24].

3. Restarting the Arnoldi Process is another approach, first described in [EE06] and detailed in [TE07]. Here, the error of the Arnoldi approximation is approximated by another Arnoldi iteration, continuing iteratively to refine the approximation.

4. Another interesting approach is to use polynomials for preconditioning the matrix A, which helps in much faster convergence of the Arnoldi process as introduced by the paper [FRHST24].

Additionally, it is worth noting that Algorithm 1 employs a modified Gram-Schmidt orthogonalization process to compute V_m, the orthonormal Krylov basis, which requires $\mathcal{O}(Nm^2)$ arithmetic operations for the Arnoldi process. This makes it computationally expensive, with increasing time requirements as N grows. In contrast, the Lanczos process requires only $\mathcal{O}(Nm)$ arithmetic operations. However, it is important to understand that the Lanczos process can only be applied to the special case.

For $f(H_m) = f(V_m^+ AV_m)$, the integral representation in FOM can be expressed as

$$f_m = \int_\Gamma \|b\| V_m (tI + H_m)^{-1} e_1 \, d\mu(t) \tag{5.4}$$

$$= \int_\Gamma x_m(t) \, d\mu(t). \tag{5.5}$$

From this, we observe that the integrand contains the FOM (or Galerkin) approximation

$$\begin{aligned} x_m(t) &:= \|b\| V_m (tI + H_m)^{-1} e_1 \\ &= V_m y_m(t) \end{aligned}$$

for the solution $x(t)$ of the shifted linear system $(tI + A)x(t) = b$. The residuals of these approximations are explicitly given by

$$r_m(t) = b - (tI + A)x(t) \tag{5.6}$$

$$= -\|b\| h_{m+1,m} (e_m^T (tI + H_m)^{-1} e_1) v_{m+1} \tag{5.7}$$

$$= \alpha(t) v_{m+1}, \tag{5.8}$$

where $\alpha(t) = -\|b\| h_{m+1,m} (e_m^T (tI + H_m)^{-1} e_1)$, and $r_m(t)$ is orthogonal to $\text{span}(V_m)$. These insights are essential in the preceding section.

5.2 Randomized Sketching For Krylov Approximations

As discussed in the previous section, the evaluation of Arnoldi approximation methods necessitates the storage of an entire Krylov basis V_m and the orthogonalization of the next Arnoldi vectors against all previous ones, which becomes problematic for large matrices. Sketching offers a potential remedy by relaxing the stringent orthogonality requirement. The proposed approach merely requires that the sketched residual $Sr_m(t)$ be orthogonal to the sketched span of the Krylov basis, $span(SV_m)$ where S is a $s \times N$ sketching matrix. This is similar to the sketched Galerkin orthogonality condition for a parametric linear system[BN19]. This requires us to have the below,

$$\hat{x_m}(t) = V_m \hat{y_m}(t) \text{ with } (SV_m)^H [Sb - S(tI + A)\hat{x_m}(t)] = 0,$$

or equivalently (if the inverted quantity is well-defined),

$$\hat{x_m}(t) = V_m \hat{y_m}(t) \text{ with } \hat{y_m}(t) = [(SV_m)^H (tSV_m + SAV_m)]^{-1} (SV_m)^H (Sb) \tag{5.9}$$

Then as mentioned in the paper [GS23], the sketched FOM approximation for $f(A)$ is defined to be,

$$\hat{f}_m := \int_\Gamma \hat{x}_m(t)\,\mathrm{d}\mu(t) = V_m \int_\Gamma \left[(SV_m)^H(tSV_m + SAV_m)\right]^{-1} \mathrm{d}\mu(t)(SV_m)^H(Sb). \qquad \text{(sFOM)}$$

Remark 5.2.1. *Some important remarks as seen in paper [GS23] are,*

1. *if $S = I \implies$ FOM and sFOM yield the same approximates.*

2. *The sketched orthogonality condition is imposed explicitly in (sFOM), hence there is no requirement for the Krylov basis V_m to be orthogonal. This means that V_m can be constructed without orthogonalization or by using a truncated orthogonalization procedure*

3. *The sketched matrices SV_m and SAV_m can be constructed on the fly during the Arnoldi iteration, being expanded by Sv_{m+1} and SAv_{m+1} when the new Krylov basis vector v_{m+1} is appended to V_m. The matrix-vector product Av_{m+1} can be reused in the following iteration so that the overall number of matrix-vector products with A remains the same as for the Arnoldi procedure without sketching.*

4. *If the full vector approximation $\hat{f}_m$ defined by (sFOM) is needed, then V_m will still need to be stored as $\hat{x}_m(t) = V_m\hat{y}_m(t)$. However, as opposed to the standard FOM approach, V_m does not need to be (fully) orthogonal and hence V_m can be held on slow memory (e.g., hard disk). Full access to V_m is only needed once the sketched FOM approximant $\hat{f}_m$ is formed, but not during the basis generation. Alternatively, the sketched approximation also makes it viable to use a two-pass approach [Bor00; SVR08] in the case of non-Hermitian A.*

5. *If only a few (say, $\ell \ll N$) selected components of $\hat{f}_m$ are needed or, more generally, a matrix-vector product $M\hat{f}_m$ with a short matrix $M \in \mathbb{C}^{\ell\times N}$, then with truncated Arnoldi only $k+1$ basis vectors v_j need to be kept in memory in addition to the small matrix MV_m.*

5.2.1 A closed formula for sketched FOM

As further investigated in the paper [GS23], if equation 5.9 is well defined, this guarantees SV_m is of full rank m and that $V_m^H S^H SV_m$ is non-singular. Re-arranging the expression inside the brackets in equation 5.9 we have,

$$\left[tV_m^H S^H SV_m + V_m^H S^H SAV_m\right]^{-1} = \left(V_m^H S^H SV_m\right)^{-1}\left[tI + V_m^H S^H SAV_m\left(V_m^T S^T SV_m\right)^{-1}\right]^{-1}$$

Hence we rewrite the sFOM approximations as,

$$\begin{aligned}\hat{f}_m &= V_m \int_\Gamma \left[tV_m^H S^H SV_m + V_m^H S^H SAV_m\right]^{-1} \mathrm{d}\mu(t)\ (SV_m)^H (Sb)\\ &= V_m (V_m^H S^H SV_m)^{-1} \int_\Gamma \left[tI + V_m^H S^H SAV_m (V_m^H S^H SV_m)^{-1}\right]^{-1} \mathrm{d}\mu(t)\ (SV_m)^H (Sb)\\ &= V_m (V_m^H S^H SV_m)^{-1} f\left(V_m^H S^H SAV_m (V_m^H S^H SV_m)^{-1}\right)(SV_m)^H (Sb). \qquad \text{(sFOM')}\end{aligned}$$

Similar to the standard FOM approximation, the closed formula for the sketched approximation (sFOM'), does not involve any integration. Moreover, sFOM and sFOM' are completely independent of the choice of V_m as long as $span(V_m) = \kappa_m(A, b)$

As elaborated in the paper [GS23], a basis whitening condition was used for their analysis without the loss of generality that the sketched basis be orthonormal for a full rank m, SV_m. The basis whitening condition is given by,

$$(SV_m)^H SV_m = I_m \qquad (5.10)$$

Thus resulting in a simpler expression,

$$\hat{f}_m = V_m f(V_m^H S^H SAV_m) V_m^H S^H Sb \qquad \text{(sFOM")}$$

If $SV_m = Q_m R_m$ is a thin QR decomposition of the (non-orthonormal) sketched basis SV_m, this could be used as a low-cost computational process rather than enforcing the basis whitening condition during the Gram–Schmidt orthonormalization process on sketched vectors. This implies we replace,

$$SV_m \leftarrow Q_m, SAV_m \leftarrow (SAV_m)R_m^{-1}, V_m \leftarrow V_m R_m^{-1} \text{ (only implicitly!)}$$

in (sFOM"), resulting in

$$\hat{f}_m = V_m (R_m^{-1} f(Q_m^H SAV_m R_m^{-1}) Q_m^H Sb \qquad \text{(sFOM"')}$$

Based on the above, a standard algorithm for sketched FOM approximation of $f(A)b$ can be represented as below.

Algorithm 3 Sketched FOM approximation of f (A)b[GS23]

Require: $A \in \mathbb{C}^{N\times N}$, $b \in \mathbb{C}^N$, function f, integers $m < s \ll N$
Ensure: $\hat{f}_m \approx f(A)b$
1: Draw sketching matrix $S \in \mathbb{C}^{s\times N}$
2: Generate (non-orthogonal) basis V_m of $K_m(A, b)$, as well as SV_m and SAV_m
3: Compute thin QR decomposition $SV_m = Q_m R_m$ {basis whitening}
4: $\hat{f}_m \leftarrow V_m \left(R_m^{-1} f\left(Q_m^H SAV_m R_m^{-1}\right) Q_m^H Sb\right)$

Remark 5.2.2. *[GS23]If SV_m and hence R_m are extremely ill-conditioned, it is better to utilize the numerical pseudoinverse instead of R_m^{-1} to reduce any numerical instability.*

5.2.2 Adaptive quadrature for sketched FOM

In the paper [GS23] to evaluate the sketched GMRES approximant (sGMRES), the integral is approximated as no closed form. For approximating the integral one can in principle use any l-point quadrature rule,

$$\int_\Gamma (tSV_m + SAV_m)^\dagger (Sb)\, \mathrm{d}\mu(t) \approx \sum_{i=1}^{\ell} w_i(t_i, SV_m + SAV_m)^\dagger (Sb) =: q_\ell(S, A, V_m, b) \tag{5.11}$$

with weights w_i and quadrature nodes $t_i \in \Gamma(i = 1, 2 \ldots, l)$. In [GS23], the author uses the paper [FGS14] as a reference and introduces a numerical quadrature. They compute the results of two quadrature rules $q_{l_1}(S, A, V_m, b)$ and $q_{l_2}(S, A, V_m, b)$ of orders $l_1 < l_2$ respectively. If,

$$||q_{l_1}(S, A, V_m, b) - q_{l_2}(S, A, V_m, b)|| < tol \tag{5.12}$$

is the absolute value of the difference between the two quadrature rules for a user-specified tolerance 'tol', we accept the result of the higher-order quadrature rule q_{l_2}. If the above equation 5.12 is not satisfied, the order of the quadrature rule is increased by setting $l_1 \leftarrow l_2$ and $l_2 \leftarrow [\sqrt{2} \cdot l_2]$. We repeat this until equation 5.12 is fulfilled.

Algorithm 4 Sketched GMRES approximation of $f(A)b$ with k-truncated Arnoldi[GS23]

Require: $A \in \mathbb{C}^{N\times N}$, $b \in \mathbb{C}^N$, function f, integers m, s, ℓ_1, ℓ_2, tolerance tol
Ensure: $\tilde{f}_m \approx f(A)b$

```
1: Draw sketching matrix S ∈ C^{s×N}
2: Generate (non-orthogonal) basis V_m of K_m(A, b), as well as SV_m and SAV_m
3: Compute thin QR decomposition SV_m = Q_m R_m {basis whitening}
4: SV_m ← Q_m, SAV_m ← (SAV_m)R_m^{-1}, V_m ← V_m R_m^{-1} {only implicitly!}
5: if contour Γ is not fixed then
6:    Compute solutions Λ of generalized rectangular EVP SAV_m x = −λ SV_m x
7:    Choose Γ such that it encircles Λ
8: end if
9: Compute quadrature rules q_ℓ1(S, A, V_m, b) and q_ℓ2(S, A, V_m, b) {see 5.11}
10: while |q_ℓ1(S, A, V_m, b) − q_ℓ2(S, A, V_m, b)| > tol do
11:    Set q_ℓ1(S, A, V_m, b) ← q_ℓ2(S, A, V_m, b) {reuse previous result}
12:    ℓ1 ← ℓ2, ℓ2 ← ℓ2 + ⌈√2 · ℓ2⌉ {increase order of quadrature rules}
13:    Compute quadrature rule q_ℓ2(S, A, V_m, b)
14: end while
15: f̃_m ← V_m q_ℓ2(S, A, V_m, b)
```

In Algorithm 4, although any quadrature rule could be used, it is necessary to emphasise that the choice of the quadrature rule should depend on f and Γ. If f is not a Stieltjes function, we are then required to additionally construct a suitable contour Γ before the numerical integration.

5.3 Polynomial preconditioning

preconditioning is one of the most well-acknowledged techniques for solving linear systems. Such a system is represented with the $f(z) = z^{-1}$ function. Let us consider a non-singular matrix M then,

$$A^{-1}b = (M^{-1}A)^{-1}M^{-1}b = M^{-1}(AM^{-1})^{-1}b \tag{5.13}$$

The equation 5.13 represents the two possible types of preconditioning. The first equality displays a left preconditioning, where we compute an approximation x_m for $A^{-1}b$ from the Krylov subspace $\mathcal{K}_m(M^{-1}A, M^{-1}b)$. The second equality leads us to the right preconditioning, where we compute the approximation $x_m = M^{-1}y_m$. Here y_m is the approximation to $(AM^{-1})^{-1}b$ from the Krylov subspace $\mathcal{K}_m(AM^{-1}, b)$.

Though we consider preconditioning to be a very useful technique, the challenge faced with this method is that we need to find the most appropriate preconditioner M, that leads us to a relatively cheaper computation of $M^{-1}u$, for any vector u. Moreover, this matrix M should bring $M^{-1}A$ / AM^{-1} closer to identity such that, this further accelerates the Krylov subspace methods to converge faster in fewer number of iterations.

In the paper [FRHST24], a proposal was introduced that enables us to borrow the idea of polynomial preconditioning of the function $f(z) = z^{-1}$ to any function f. The property of interest in the function $f(z) = z^{-1}$ is that,

$$(z_1 z_2)^{-1} = z_1^{-1} z_2^{-1} = z_2^{-1} z_1^{-1}$$

If we translate this to matrix functions for any two non-singular matrices A and B we have,

$$(AB)^{-1} = B^{-1}A^{-1} = A^{-1}B^{-1}$$

This is what was reflected in the equation 5.13. This property is representable only if A and B commute, in which $f(A)g(B) = g(B)f(A)$ for any functions f and g, thus in particular for $f(z) = g(z) = z^{-1}$. Now to implement this idea the paper [FRHST24] suggests identifying the situation where $f(AB)$ can be smoothly interlinked to $f(A)$ and/or $f(B)$. The proposal made in the paper is that, assuming $A \in \mathbb{C}^{n\times n}$ with a polynomial p and a function z^α for some $\alpha \in \mathbb{R}$ where, $f(z) = g(z) = z^\alpha$. Furthermore, if $\alpha < 0$ an assumption is made such that matrices A and $p(A)$ do not have eigenvalues in $(-\infty, 0]$. Then,

$$(Ap(A))^\alpha = A^\alpha (p(A))^\alpha = (p(A))^\alpha A^\alpha \tag{5.14}$$

5.3.1 Preconditioning for inverse square root

The approach introduced in the polynomial preconditioning method for the inverse square root is to,

1. approximate $p(A)$ as close as possible to A^{-1} such that $Ap(A)$ is very close to identity.
2. $(p(A))^{1/2}$ needs to be easily evaluated.

To achieve these goals, the paper[FRHST24] suggests to consider $p(z) = (q(z))^2$, where q is chosen as a polynomial that approximates $z^{-1/2}$. Modifying the equation 5.14 for $\alpha = -1/2$ gives,

$$A^{-1/2}b = (A(q(A))^2)^{-1/2}q(A)b = q(A)(A(q(A))^2)^{-1/2}b \tag{5.15}$$

where we know,

$$((q(A))^2)^{-1/2} = q(A). \tag{5.16}$$

Satisfying the equation 5.16 directly correlates to the branch we consider for the square root and the distribution of the eigenvalues of A. In paper [FRHST24] a further assumption is made where only the principle branch of the square root is considered i.e.,

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

i.e., the branch cut is put on the negative real axis. Hence for any polynomial q we have,

$$((q(A))^2)^{-1/2} = q(A) \text{ if } \operatorname{spec}(q(A)) \in \mathbb{C}^+ \tag{5.17}$$

where $\mathbb{C}^+$ denotes the open right half-plane.

As a result of the above implications if $q(A)$ approximates $A^{-1/2}$, the matrix $A(q(A))^2$ should be close to identity and thus have a small condition number. This signifies we require only fewer iterations for obtaining a more accurate approximation f_m

Algorithm 5 m steps of left polynomially preconditioned Arnoldi for $A^{-1/2}b$ [FRHST24]

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

Algorithm 6 m steps of right polynomially preconditioned Arnoldi for $A^{-1/2}b$ [FRHST24]

Require: Polynomial q such that $q(A)$ approximates $A^{-1/2}$
Ensure: $f_m \leftarrow Y_m(H_m^{-1/2}e_1\|b\|)$

```
Choose polynomial q such that q(A) approximates A^{-1/2}
Put v_1 ← b/||b||
for j = 1, ..., m do
    {Arnoldi process}
    Compute y_j ← q(A)v_j, u ← q(A)y_j, w ← Au
    for i = 1, ..., j do
        h_ij ← ⟨w, v_i⟩, w ← w − v_i h_ij {orthogonalize against previous vectors}
    end for
    h_{j+1,j} ← ||w||
    v_{j+1} ← w/h_{j+1,j}
end for
f_m ← Y_m(H_m^{-1/2} e_1 ||b||), Y_m = [y_1 ... y_m], H_m = (h_ij) ∈ C^{m×m} {upper Hessenberg}
```

While comparing both the above two algorithms, with left preconditioning, the norm $||f_m||$ can be obtained just from $H^{-1/2}e_1||b||$ since V_m is orthonormal. But for the right preconditioning, Y_m does not have orthonormal columns. Hence when basing a stopping criteria on the size of the difference of consecutive iterates, left preconditioning is usually more appropriate.

Though a proposal has been established for the polynomial preconditioning, another difficult task for using the algorithms is the selection of the polynomial based on the properties of the matrix A. The paper [FRHST24] elaborates to what extent the equation 5.17 is fulfilled for the polynomial chosen. i.e., $((q(A))^2)^{1/2}$. Thus a general result has been used for the choice of the polynomial. Assume $\operatorname{spec}(A) \subseteq \mathbb{C}^+$ and that q approximates $z^{-1/2}$ on $\operatorname{spec}(A)$ uniformly in a relative sense with accurately $\frac{1}{\sqrt{z}}$. i.e., we have,

$$|q(\lambda) - \lambda^{-1/2}| \leq \frac{1}{\sqrt{2}}|\lambda^{-1/2}| \quad \text{for} \quad \lambda \in \operatorname{spec}(A)$$

Then $((q(A))^2)^{-1/2} = q(A)$. Based on the above-stated fulfilment criteria, some interesting choices of polynomials studied in the paper were Chebyshev expansions, polynomial interpolation at (harmonic) Ritz values, and polynomials obtained via error via minimization.

In this thesis we are interested in obtaining polynomial interpolation at Ritz values, utilizing the Arnoldi method. The Ritz values are the eigenvalues of the upper Hessenberg matrix H_d arising from d steps of the Arnoldi process. Hence a simple idea to choose the preconditioning polynomial q is as the polynomial of degree $d-1$ that interpolates $\frac{1}{\sqrt{z}}$ at the d Ritz values. This has the attractive feature that it does not require any prior knowledge about the spectral region of A but rather adapts itself automatically to the spectrum of A. For, the Arnoldi process for constructing H_d one can start with a randomly drawn vector. Once we have the interpolation points one could use different bases to represent the interpolating polynomial. A very widely used representation is the Newton's representation,

$$P_{m-1}(\alpha) = \sum_{i=1}^{m} a_i \prod_{j=1}^{i-1} (\alpha - \theta_j), \tag{5.18}$$

and to have the polynomial p interpolating on the points θ_i, we need to use divided differences to obtain the coefficients a_i , as follows:

$$\begin{aligned}
a_1 &= f[\theta_1] = f(\theta_1) \\
a_2 &= f[\theta_1, \theta_2] = \frac{f(\theta_2) - f(\theta_1)}{\theta_2 - \theta_1} \\
a_3 &= f[\theta_1, \theta_2, \theta_3] = \frac{f[\theta_2, \theta_3] - f[\theta_1, \theta_2]}{\theta_3 - \theta_1} \\
&\vdots \\
a_m &= f[\theta_1, \theta_2, \theta_3, \ldots, \theta_m] = \frac{f[\theta_2, \theta_3, \ldots, \theta_m] - f[\theta_1, \theta_2, \ldots, \theta_{m-1}]}{\theta_m - \theta_1}
\end{aligned}$$

where, $f[\theta_2, \theta_3] = \frac{f(\theta_3) - f(\theta_2)}{\theta_3 - \theta_2}$.

5.4 Restarted Arnoldi

One of the most useful definitions for the Arnoldi approximation is,

$$f_m = V_m f(H_m) V_m^H b = ||b|| V_m f(H_m) e_1 \tag{5.19}$$

This provides a clear understanding of the difference between the Arnoldi approximation and polynomial interpolation. This is exploited in many methods to approximate $f(A)b$. Even though the above relationship is helpful, there are two main problems one encounters.

The first problem is the computation and the storage of the whole Arnoldi basis V_m for the evaluation of 5.19. This becomes expensive with the growth of m to a large number. The second problem faced is that $f(H_m)e_1$ has to be computed for forming f_m. This could also get expensive as m grows to a bigger number. For a Hermitian matrix A, a simple strategy to overcome the storage problem would be a two-pass Lanczos method [SVR08]. Now in the case of a non-Hermitian matrix A, which is of our interest, a suggestible solution to the above problem would be restarted Arnoldi. With this method, the m Arnoldi orthogonalization steps are carried out to form f_m. The basis V_m computed thereafter gets discarded and a second cycle of Arnoldi is undergone. The Arnoldi cycle is restarted to approximate the error $d_m = x - f_m$ where x denotes the sought solution of the linear system $Ax = b$.

The above restart procedure is possible because the error d_m solves the residual equation,

$$Ad_m = r_m \tag{5.20}$$

and the residual $r_m = b - Af_m$.

As mentioned in the paper [FGS14] for the development of a restarted technique for a general function f, the challenge faced is directly correlated to the residual equation.

However, as cited in the papers[Boo; EE06], it is possible to introduce the error of the restarted Arnoldi approximations via divided differences. As stated by Eiermann and Ernst in their paper[EE06], assuming that we have $A \in \mathbb{C}^{N\times N}$, $b \in \mathbb{C}^N$, with Arnoldi-like decomposition, $AV_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^T$ and $w_m(z) = (z-\theta_1)\dots(z-\theta_m)$ is the nodal polynomial associated with Ritz values $\theta_1 \dots \theta_m$, are the eigenvalues of H_m then, the error of f_m defined in 5.19 is given by,

$$f(A)b - f_m = ||b||\gamma_m[D_{w_m} f](A)v_{m+1} =: e_m(A)v_{m+1} \tag{5.21}$$

where $[D_{w_m} f]$ denotes the m-th divided difference of f with respect to the interpolation nodes $\theta_1 \dots \theta_m$ and $\gamma_m = \prod_{i=1}^{m} h_{i+1,i}$. This helps us represent error after m steps of the Arnoldi method which could be used to perform restarts similar to the linear system. A summarized algorithm for the above form of restart is as given below:

Algorithm 7 Restarted Arnoldi method for $f(A)b$ from [FGS14] (generic version).

Require: A, b, f, m

1: Compute the Arnoldi decomposition $AV_m^{(1)} = V_m^{(1)} H_m^{(1)} + h_{m+1,m}^{(1)} v_{m+1}^{(1)} e_m^T$ with respect to A and b.
2: Set $f_m^{(1)} := \|b\| V_m^{(1)} f(H_m^{(1)}) e_1$.
3: **for** $k = 2, 3, \dots$ **until convergence do**
4: Determine the error function $e_m^{(k-1)}(z)$.
5: Compute the Arnoldi decomposition $AV_m^{(k)} = V_m^{(k)} H_m^{(k)} + h_{m+1,m}^{(k)} v_{m+1}^{(k)} e_m^T$ with respect to A and $v_{m+1}^{(k-1)}$.
6: Set $f_m^{(k)} := f_m^{(k-1)} + \|b\| V_m^{(k)} e_m^{(k-1)}(H_m^{(k)}) e_1$.
7: **end for**

However as shown in the paper, combining the error representation with the above algorithm provides a restarted Arnoldi, but it is not practically feasible due to numerical instabilities. The problem arises since in divided differences, the evaluation of high-order divided differences is prone to instabilities, due to the interpolation nodes being close to each other, thereby causing subtractive cancellations and very small denominations in the divided difference table. In the case of the Hermitian matrix A a different approach was also investigated as cited in [ITS09]. Here along with the prior mentioned error representation an assumption that A is hermitian is made. Let w_m be a unitary matrix whose columns are the eigenvectors of H_m and $\alpha_i = e_1^T w_m e_i$, $(i = 1, \dots, m)$. Then an improved error representation can be defined as below:

$$f(A)b - f_m = ||b||h_{m+1,m} g(A) v_{m+1} \tag{5.22}$$

with

$$g(z) = \sum_{i=1}^{m} \alpha_i \gamma_i D_{w_i}(z) \quad \text{where} \quad w_i(z) = (z - \theta_i) \tag{5.23}$$

The error representation 5.22 involves only first-order divided differences, thus making it less prone to numerical instabilities. Yet, this method is stable to a limited extent as mentioned in paper[ITS09].

The original restart method [EE06] is highly unstable. Hence an alternative approach without the use of an error function needs to be taken into consideration. One way would be to use the same function f throughout all the restart cycles. This is realized because the Arnoldi-like approach approximates from consecutive cycles to satisfy the update,

$$f_m^{(k)} = f_m^{(k-1)} + ||b|| V_m^{(k)} \left[f(H_{km}) e_1 \right]_{(k-1)m+1:km}, \quad k \geq 2 \tag{5.24}$$

where H_m is the accumulation of all the Hessenberg matrices from the previous rest cycles in a block-Hessenberg matrix form. i.e.,

$$H_{km} = \begin{pmatrix} H_{(k-1)m} & O \\ h_{m+1,m} e_1 e_{(k-1)m}^T & H_m^{(k)} \end{pmatrix} \tag{5.25}$$

In the newly updated form of approximation of f_m, presented in 5.24, the stability problems that were raised in the previous implementation have been sorted out. This also resolves the price of storage on the Arnoldi basis. These perks come however at the cost of evaluating f on the Hessenberg matrix that increases its size by km (i.e., the computational cost grows cubically in km). The Algorithm for the improved restart methods is as follows:

Algorithm 8 Restarted Arnoldi approximation for $f(A)b$ from [AEEG08].

Require: A, b, m, rational approximation $r \approx f$ of the form 5.26
1: Set $f_m^{(0)} = 0$ and $v_{m+1}^{(0)} = b$.
2: **for** $k = 1, 2, \ldots$ **until convergence do**
3: Compute the Arnoldi decomposition $AV_m^{(k)} = V_m^{(k)} H_m^{(k)} + h_{m+1,m}^{(k)} v_{m+1}^{(k)} e_m^T$ with respect to A and $v_{m+1}^{(k-1)}$.
4: **if** $k = 1$ **then**
5: **for** $i = 1, \ldots, \ell$ **do**
6: Solve $(t_i I - H_m^{(k)}) r_{i,1} = e_1$.
7: **end for**
8: **else**
9: **for** $i = 1, \ldots, \ell$ **do**
10: Solve $(t_i I - H_m^{(k)}) r_{i,k} = h_{m+1,m}^{(k-1)} (e_m^T r_{i,k-1}) e_1$.
11: **end for**
12: **end if**
13: $h_m^{(k)} = \sum_{i=1}^{\ell} \alpha_i r_{i,k}$.
14: Set $f_m^{(k)} := f_m^{(k-1)} + \|b\| V_m^{(k)} h_m^{(k)}$.
15: **end for**

The rational function in the above algorithm is given by,

$$r(z) = \sum_{i=1}^{l} \frac{\alpha_i}{t_i - z} \tag{5.26}$$

Then it can be seen that the evaluation of 5.24 with $f = r$ is possible at a constant work per restart cycle. Evaluating $(t_i I - H_{km})^{-1} e_1$ via sequential solution of k shifted linear system,

$$(t_i I - H_m^{(1)}) r_{i,1} = e_1, \tag{5.27}$$

$$(t_i I - H_m^{(j)}) r_{i,j} = h_{m+1,m}^{(j-1)} (e_m^T r_{i,j-1}) e_1, \quad j = 2, \ldots, k \tag{5.28}$$

It can be observed that the exploitation of the last block of $r(H_{km})e_1$ is only required. Thus allowing an efficient restarting for general functions f with the closure that sufficiently accurate rational approximation r (with $r(A)b \approx f(A)$ is available.

5.4.1 Error function in integral form

The main problem with the algorithm 7 was the divided difference. The paper [FGS14] introduces a new proposal where the error functions are evaluated using their corresponding integral representation rather than applying the divided differences along with conditions to be fulfilled to do so. The paper introduces a formula for the interpolating polynomials of functions that are representable as a Cauchy type integral i.e.,

$$f(z) = \int_\Gamma \frac{g(t)}{t - z} \, dt, \quad z \in \Omega \tag{5.29}$$

where $\Omega \subset \mathbb{C}$ is a region, $f : \Omega \to \mathbb{C}$ is analytic with the path $\Gamma \subset \mathbb{C} \setminus \Omega$ and $g : \Gamma \to \mathbb{C}$. If the integral exists then we can write the interpolation polynomial p_{m-1} of f with the interpolation nodes $\theta_1, \ldots, \theta_m \subset \Omega$ as,

$$p_{m-1}(z) = \int_\Gamma \left(1 - \frac{w_m(z)}{w_m(t)}\right) \frac{g(t)}{t - z} dt \tag{5.30}$$

where $w_m(z) = (z - \theta_1) \ldots (z - \theta_m)$. With the above as the foundation, [FGS14] further investigates two important scenarios:

1. f is holomorphic on a region $\Omega' \supset \Omega$.
2. f is a Stieltjes function.

Theorem 5.4.1. *[FGS14] Let f have an integral representation 5.29 and let $A \in \mathbb{C}^{N \times N}$ with spec$(A) \subset \Omega$ and $b \in \mathbb{C}^N$ be given. Denote by f_m the m-th Arnoldi approximation 5.19 to $f(A)b$ with spec$(H_m) = \{\theta_1, \ldots, \theta_m\} \subset \Omega$. Then provided that the integral 5.29 with $w_m(t) = (t - \theta_1)$ůůů$(t - \theta_m)$ exists,*

$$f(A)b - f_m = \gamma_m \int_\Gamma \frac{g(t)}{w_m(t)} (tI - A)^{-1} v_{m+1} \, dt =: e_m(A) v_{m+1} \tag{5.31}$$

where $\gamma_m = \prod_{i=1}^m h_{i+1,i}$.

The above theorem is very useful as it shows how to interpret the error of the Arnoldi approximation f_m to the error of $f(A)b$ approximation represented by $e_m(A)$ applied to a vector. This could be used as a substitute to divide differences in different algorithms which had numerical instabilities. Hence we can extend further the integral representation to subsequent restart cycles due to the general form of the Cauchy-type integral being adopted. i.e., the error of $f_m^{(k)}$ satisfies,

$$f(A)b - f_m^{(k)} = \gamma_m^{(1)} \cdots \gamma_m^{(k)} \int_\Gamma \frac{g(t)}{w_m^{(1)}(t) \cdots w_m^{(k)}} (tI - A)^{-1} v_{m+1}^{(k)} dt =: e_m^{(k)}(A) v_{m+1}^{(k)} \tag{5.32}$$

provided that the integral 5.29 with $w_m(t) = (w_m^{(1)}(t) \ldots {}_m^{(k)}(t))$ exists.

Here we are more interested in Stieltjes functions as they suit our requirement with the sign function. Moreover, Stieltjes functions ensure the existence of the integral which is a vital part of the integral representation of the error functions. The path these functions have, $\Gamma = (-\infty, 0]$ is fixed and does not depend on the spectrum of A. An example of Stieltjes function that is also of interest to us is,

$$f(z) = z^{-\alpha} = \frac{\sin((\alpha - 1)\pi)}{\pi} \int_{-\infty}^{0} \frac{(-t)^{-\alpha}}{t - z} dt \quad \text{for } \alpha \in (0, 1) \tag{5.33}$$

Since the path of the Stieltjes function Γ is in the real interval, one can find an elegant integral transformation to approximate the infinite integral with a numerical quadrature method.

5.4.2 Evaluation of the error function by numerical quadrature

The paper [FGS14], exploits the ability to approximate the action of the error function $e_m(A)b$, which is used in restarting the Arnoldi process, adopting numerical quadrature to approximate the integral 5.31. A typical choice of a suitable form of quadrature formula is,

$$\hat{e}_m(z) = \gamma_m \sum_{i=1}^{l} w_i \frac{g(t_i)}{w_m(t_i)} \frac{1}{t_i - z} \tag{5.34}$$

with quadrature nodes $l_i \in \Gamma$ and weights w_i. From equation 5.34 it is observed that it is a rational approximation. Thus the approach is very similar to Algorithm 8. Assume that the quadrature nodes and the weights in 5.34 are fixed throughout every restart cycle in Algorithm 7. Also if Algorithm 8 utilizes a rational approximation of the form 5.26 with poles t_i and weights $\alpha_i = w_i g(t_i)$. Let the quadrature formula be used to evaluate f in the first restart cycle of Algorithm 7. These assumptions make both the algorithms mathematically equivalent at each restart for $k \geq 1$.

The newly introduced quadrature-based restart approach has several perks over the other two restart methods.

1. For constructing a fixed rational approximant r where $r(A)b \approx f(A)b$ in Algorithm 8, an a-prior information on the spectrum of A is necessary. In the integral approach, error 5.32 allows the automated construction of rational approximations without any spectral data (given the path Γ does not depend on the spectrum of A). Thus providing an option to apply over a broader range of applications.

2. The same rational approximations are used by Algorithm 6 for every restart cycle. The vector $r_{i,k}$ in 5.27 and 5.28 are hence needed to be stored and updated separately for each elementary shifted linear system. On the other hand, the integral representation approach does not require a fixed quadrature rule 5.34). It can be dynamically adapted in each restart cycle to evaluate $e_m^{(k-1)}(H_m^k)e_1$ with the required accuracy.

3. The quadrature allows adaptivity and error control inherently.

The generic way of implementing an algorithm is as below:

Algorithm 9 Quadrature-based restarted Arnoldi approximation for $f(A)b$ [FGS14]

Given: A, b, f, m, tol.

Compute the Arnoldi decomposition $AV_m^{(1)} = V_m^{(1)}H_m^{(1)} + h_{m+1,m}^{(1)}v_{m+1}^{(1)}e_m^T$ with respect to A and b.
Set $f_m^{(1)} := \|b\|V_m^{(1)}f(H_m^{(1)})e_1$.
Set $\tilde{\ell} := 8$ and $\ell := \text{round}(\sqrt{2}\cdot\ell)$.
for $k = 2, 3, \ldots$ until convergence **do**
    Compute the Arnoldi decomposition $AV_m^{(k)} = V_m^{(k)}H_m^{(k)} + h_{m+1,m}^{(k)}v_{m+1}^{(k)}e_m^T$ with respect to A and $v_{m+1}^{(k-1)}$.
    Choose sets $(t_i, \omega_i)_{i=1,\ldots,\tilde{\ell}}$ and $(t_i, \omega_i)_{i=1,\ldots,\ell}$ of quadrature nodes/weights.
    Set `accurate` := `false` and `refined` := `false`.
    while `accurate` = `false` **do**
        Compute $\tilde{h}_m^{(k)} = (e_m^{(k-1)})^T f(H_m^{(k)})e_1$ by quadrature of order $\tilde{\ell}$.
        Compute $h_m^{(k)} = (e_m^{(k-1)})^T f(H_m^{(k)})e_1$ by quadrature of order ℓ.
        if $\|h_m^{(k)} - \tilde{h}_m^{(k)}\| < tol$ **then**
            `accurate` := `true`.
        else
            Set $\tilde{\ell} := \ell$ and $\ell := \text{round}(\sqrt{2}\cdot\tilde{\ell})$.
            Set `refined` := `true`.
        end if
    end while
    Set $f_m^{(k)} := f_m^{(k-1)} + \|b\|\|V_m^{(k)}h_m^{(k)}\|$.
    if `refined` = `false` **then**
        Set $\tilde{\ell} := \ell$ and $\ell := \text{round}(\ell/\sqrt{2})$.
    end if
end for

Here the use of adaptive quadrature is seen. At each restart, the integral of the error function is approximated with a different number of quadrature nodes $\tilde{l}$ and $l(\tilde{l} < l)$.Moreover, the paper [FGS14] suggests the possibility of introducing deflation to the above Algorithm 9 presented in [EEG11]. To achieve this, after every restart cycle k, reordering of the Schur decomposition of $H_m^{(k)}$ is done to restart the Arnoldi process with a set of d target Ritz vectors. The paper also presents some examples of functions that implement Algorithm

9, where insights were discussed on the integral transformation function and the suitable selection of quadrature rules based on the function under consideration.

The inverse fractional powers of $f(z) = z^{-\alpha}$ for $\alpha \in (0,1)$, the function of importance in this thesis, are Stieltjes functions. This provides the advantage that the path Γ is always explicitly known and is independent of the spectrum of A. However, this comes with a demerit of dealing with infinite integration intervals. As per the paper [Gau91], one approach to overcome this hurdle would be the introduction of Gaussian quadrature rules for infinite integration intervals. Another approach would be the application of variable substitution and transforming the infinite integral to a finite integral based on [Car12] working with integral representation for the matrix p-th root.

Lemma 5.4.2. *[FGS14]Let $z \in \mathbb{C} \setminus \mathbb{R}^-$. Then for all $\beta > 0$*

$$z^{-\alpha} = \frac{2\sin((\alpha-1)\pi)\beta^{1-\alpha}}{\pi} \int_{-1}^{1} \frac{(1-x)^{-\alpha}(x+1)^{\alpha-1}}{-\beta(1-x)-z(1+x)} dx \tag{5.35}$$

Lemma 5.4.3. *[FGS14]Let $\beta > 0$ and let x_i and $\omega_i (i = 1, ..., l)$ be the nodes and weights of the l-node Gauss-Jacobi quadrature rule on $[-1,1]$. Then*

$$r_{l-1,l}(z) = \frac{2\sin((\alpha-1)\pi)\beta^{1-\alpha}}{\pi} \sum_{i=1}^{l} \frac{\omega_i}{-\beta(1-x_i)-z(1+x_i)} \tag{5.36}$$

is the $(l-1,l)$-Padé approximant for $z^{-\alpha}$ with expansion point β.

The above lemma suggests that if the spectrum of A is clustered around β, then the rational approximation 5.36 is well suited for $A^{-\alpha}$. A reasonable choice of transformation parameter, $\beta = \frac{trace(A)}{n}$, the arithmetic mean of eigenvalues of A. The numerical experiments presented in the paper [FGS14], suggest that the method is not very sensitive to the choice of β. The disadvantage of random choice of β is the increase in the number of quadrature nodes l required for the computation, which shoots up the computational cost.

The discussion done until now of using the quadrature rule was on the original function $f(z) = z^{-\alpha}$. However, the application of the same on the error function $e_m(z)$ is more appealing in terms of the algorithm 9. In this situation, the insertion of Cayley transforms $t = -\beta\frac{1-x}{1+x}$ to 5.32 and the integral representation 5.33 of $z^{-\alpha}$ leads to the error function,

$$\frac{2\sin((\alpha-1)\pi)\beta^{1-\alpha}\gamma_m}{\pi} \int_{-1}^{1} \frac{1}{w_m\left(-\beta\frac{1-x}{1+x}\right)} \frac{(1-x)^{-\alpha}(x+1)^{\alpha-1}}{-\beta(1-x)-z(1+x)} dx \tag{5.37}$$

The Gauss-Jacobi quadrature can handle the singularities at the endpoints of the interval $[-1,1]$. But the term, $\frac{1}{w_m}(-\beta\frac{1-x}{1+x})$ introduces m additional singularities in the integrated; see the prior definition of $w_m(z) = (z-\theta_1)\ldots(z-\theta_m)$ of the nodal polynomial. This means that the singularities of the non-transformed integrand are the Ritz values. They could lie anywhere in the field of values of A, in or out of the integration. Thus one can only guarantee that there are no singularities on the interval of integration if the field of values of A is disjoint from the negative real axis.

6 Deflation

While evaluating the sign functions for the specific case of hermitian matrices, it is well proven that deflating the eigenvalues from the smallest could act as a catalyst to accelerate the computation [EFL+02]. The reason behind utilizing this is crucial since the sign function is discontinuous at zero. This is analogous to the Non-Hermitian matrices as they have a discontinuity along the imaginary axis. A solution to the above problem would be to approximate f at the eigenvalues of A by a low-order polynomial. However, suppose the gap between the eigenvalues of A to the left and right of the imaginary axis is too small. In that case, there exists no low-order polynomial accurate for all the eigenvalues.

Deflation introduces the idea that we dissect these critical eigenvalues from the rest and solve them exactly. Krylov subspace methods approximate the remaining deflated space. In the Hermitian case, deflation is straightforward since eigenvectors are orthonormal as mentioned in [BFLW07]. For the non-Hermitian matrices, we are interested in, the (generalized) eigenvectors which are not orthonormal. For this reason, the spectral definition of the matrix functions cannot be easily decomposed into orthogonal subspaces as the definition involves the inverse of the matrix of the basis vectors. The paper[BFLW07] introduces some proposals to overcome this scenario using the composite subspace generated by including a small number of critical eigenvalues in the Krylov subspace.

6.1 LR-deflation

In this approach an augmented subspace $\Omega_m + \mathcal{K}_m(A, x)$ is constructed using both left and right eigenvectors in respect to the critical eigenvalues. Here, the m critical eigenvalues and the left and right eigenvectors of A can be computed using appropriate iterative methods. The right eigenvectors satisfy,

$$AR_m = R_m \lambda_m \tag{6.1}$$

In the above equation, λ_m is the diagonal eigenvalue matrix for m critical eigenvalues. $R_m = [r_1, \ldots, r_m]$ the matrix of the right eigenvectors (stored as columns). The left eigenvectors satisfy,

$$L_m^\dagger A = \lambda_m L_m^\dagger \tag{6.2}$$

Here $L_m = [_1, \ldots, l_m]$ is the matrix containing the left eigenvectors (stored in columns). In a non-Hermitian matrix, the left and right eigenvectors corresponding to different eigenvalues are orthogonal. If there exist degenerate eigenvalues, then linear combinations of the eigenvectors can be formed in such a way that the orthogonality property remains in general valid. For normalized eigenvectors $L_m^\dagger R_m = I_m$ i.e., $l_i^\dagger r_i = 1$. Furthermore, $R_m L_m^\dagger$ is an oblique projector on the subspace Ω_m, spanned by the right eigenvectors.

Based on the above details we can now decompose the vector x as,

$$x = x_\parallel + x_\ominus \tag{6.3}$$

where $x_{||} = R_m L_m^{\dagger} x$, the oblique projection of x_m on Ω_m and $x_{\ominus} = x - x_{||}$. Now applying the decomposition of x from the equation 6.3 on $f(A)$ yields,

$$\begin{aligned} f(A)x &= f(A)x_{||} + f(A)x_{\ominus} \\ &= f(A)R_m L_m^{\dagger} x + f(A)x_{\ominus} \end{aligned} \tag{6.4}$$

Now as per the previously introduced idea, the first part of the equation 6.4 could be evaluated exactly using the spectral definition of the matrix functions i.e.,

$$f(A)R_m L_m^{\dagger} x = R_m f(\lambda_m) L_m^{\dagger} x \tag{6.5}$$

The second term could be approximated with the help of some Krylov subspace approaches. This means an orthonormal basis is constructed in the Krylov subspace $\mathcal{K}_k(A, x_{\ominus})$. For the Arnoldi method, the subspace is created using the recurrence,

$$AV_k = V_k H_k + \beta_k v_{k+1} e_k^T \tag{6.6}$$

where, $\beta = |x_{\ominus}|$ and $v_1 = \frac{x_{\ominus}}{\beta}$. The main advantage of such a deconstruction of x to two components is that we could separate the effects of the critical eigendirections for a better approximation of the action of a matrix over a vector i.e., $\mathcal{K}_k(A, x_{\ominus})$ does not mix with Ω_m. The above is summarized in the form of an algorithm as below:

Algorithm 10 Algorithm for approximating $f(A)x$ in the LR-deflation scheme [BFLW07]

Given: Matrix A, vector x, and function f.
Output: Approximation of $f(A)x$.

1: Determine the left and right eigenvectors for m critical eigenvalues of A using ARPACK. Store the corresponding eigenvector matrices L_m and R_m.
2: Compute $f(\lambda_i)$ for $i = 1, \ldots, m$ for the critical eigenvalues.
3: Compute $x_{\Theta} = \left(1 - R_m L_m^{\dagger}\right) x$.
4: Construct an orthonormal basis for the Krylov subspace $\mathcal{K}_k(A, x_{\Theta})$ using the Arnoldi recurrence. The basis is constructed iteratively by orthogonalizing each new Krylov vector for all previous Arnoldi vectors and is stored as columns of a matrix V_k. Also, build the upper Hessenberg matrix $H_k = V_k^{\dagger} A V_k$.
5: Compute the (first column of) $f(H_k)$.
6: Compute the approximation to $f(A)x$ using 6.4.

7 Exploration of Possibilities

So far, we have explored various approaches currently present for computing the sign function of a non-Hermitian matrix. Each method analyzed has its own advantages depending on the specific computational context. In this Chapter, we aim to combine various methods to leverage their respective strengths and enhance the overall computation.

7.1 Combination of LR-deflation with Krylov Methods

In Chapter 6, we discussed the paper [BFLW07], which demonstrates that deflation can potentially serve as an accelerator. There we were introduced to the concept of a composite subspace. Building on this foundation, our proposed combination of methods incorporates these ideas and integrates efficient Krylov subspace approaches to further enhance computational performance. In our study, we specifically adopted the LR-deflation approach, as numerical experiments in [BFLW07] demonstrate that the LR-deflation scheme offers significantly better accuracy and requires less CPU time per iteration compared to other deflation methods the paper tested. As we are considering LR-deflation as the base for our combinations for evaluation, let's recap the method outlining the important ingredients to develop the new methods.

The key idea behind LR-deflation is the construction of an augmented subspace, $\Omega_m + \mathcal{K}_m(A, x)$, which incorporates both left and right eigenvectors for m critical eigenvalues. As discussed in the paper [BFLW07], degenerate eigenvalues can be addressed by forming linear combinations of eigenvectors, thereby preserving orthogonality in a generalized sense. This allows us to create an oblique projector $R_m L_m^\dagger$ onto the subspace Ω_m, spanned by the right eigenvectors.

With these results, the vector x can be decomposed within the composite subspace as follows:

$$\begin{aligned} x &= x_\parallel + x_\ominus, \\ x_\parallel &= R_m L_m^\dagger x, \\ x_\ominus &= x - x_\mathcal{K}, \end{aligned}$$

where $x_\parallel$ represents the oblique projection of x onto Ω_m. Substituting this into the action of a matrix, the expression for $f(A)x$ can be rewritten as:

$$\begin{aligned} f(A)x &= f(A)(x_\parallel + x_\ominus) \\ &= f(A)x_\parallel + f(A)x_\ominus \\ &= f(A)R_m L_m^\dagger x + f(A)x_\ominus. \end{aligned}$$

Applying the spectral definition of the matrix function, we obtain:

$$f(A)R_m L_m^\dagger x = R_m f(\lambda_m) L_m^\dagger x.$$

where, λ_m is the diagonal eigenvalue matrix for m critical eigenvalues.

Our primary interest in creating new combinations lies in evaluating $f(A)x_\ominus$. As discussed in the paper [BFLW07], $f(A)x_\ominus$ can be computed efficiently using appropriate Krylov subspace methods depending on the application. However, the study presented in the paper[BFLW07] was limited to just the implementation of Arnoldi iteration combined with LR-deflation.

A general algorithm for combining Krylov methods with LR-deflation can be outlined as follows:

Algorithm 11 Framework for Approximating $f(A)x$ using a Combination of LR-Deflation and Krylov Subspace Methods

---

Given: Matrix A, vector x, function f and no. of deflated eigenvectors m.
Output: Approximation of $f(A)x$.

1: Determine the left and right eigenvectors for m critical eigenvalues of A. Store the corresponding eigenvector matrices L_m and R_m.
2: Compute $f(\lambda_i)$ for $i = 1, \ldots, m$ for the critical eigenvalues.
3: Compute $x_\Theta = \left(1 - R_m L_m^\dagger\right) x$.
4: Approximant for $f(A)x_\ominus$ is computed using Krylov subspace method of your interest.
5: Compute the approximation to $f(A)x$ using 6.4.

The choice of LR-deflation over other methods in our study is motivated by the following factors [BFLW07]:

1. Unlike other deflation methods such as the Schur deflation, the absence of coupling between subspaces.

2. Krylov subspace methods do not require the deflated directions to be (obliquely) projected out of the Krylov subspace, as the subspaces remain distinct.

$$A(I - R_m L_m^\dagger) = (I - R_m L_m^\dagger)A \tag{7.1}$$

However, no method is without its limitations. While LR-deflation offers the above advantages, it comes with the drawback of a longer initial phase of computation, as it requires the calculation of both left and right eigenvectors.

7.1.1 Rationale for the Selection of Methods in the Combination

From the perspective of our application, specifically, the QCD lattice problem involving non-Hermitian matrices, several Krylov methods could benefit from acceleration through LR-deflation. After reviewing various approaches, we identified a few methods that are particularly well-suited to the problem at hand. These include:

1. Quadrature-based restarted Arnoldi method,

2. Polynomial preconditioning method,
3. Quadrature-based sketched FOM.

These methods and their corresponding algorithms were thoroughly discussed in Chapter 5. They can be seamlessly integrated into the LR-deflation framework 11 at the stage where $f(A)x_{\ominus}$ is evaluated, thereby enhancing the overall computation of the approximate. In this section, we present a few justifications for the chosen methods, explaining why these specific combinations are of interest for computing the action of the sign function of a matrix on a vector.

7.1.1.1 Quadrature-based restarted Arnoldi method

In reviewing the paper on quadrature-based restarted Arnoldi [FGS14], the authors support their method through numerical experiments demonstrating its stability and efficiency—two highly desirable properties for Krylov methods. Additionally, the method is particularly significant due to its ability to limit memory usage through restarts, where, at each restart, the last Krylov basis vector is used as the new initial residual vector.

The plots presented in the paper [FGS14], which illustrate the absolute 2-norm error over cycles, further demonstrate the method's superiority in convergence compared to divided difference and rational approximation methods. Additionally, the paper reports promising numerical experiments with both restarted explicit and implicit deflation, as evidenced by the plot of absolute 2-norm error over cycles. Notably, these numerical experiments were conducted on the same application we are addressing, further reinforcing the suitability of the Quadrature-based restarted Arnoldi method for our study.

7.1.1.2 Polynomial preconditioning method

The authors of the paper [FRHST24] on polynomial preconditioned Arnoldi discussed the effects of polynomial preconditioning on the spectrum of the matrix. They specifically investigated the case of a Hermitian positive definite matrix A, which serves as a reflection of more general settings. They measured the quality of the preconditioner, which depends on the accuracy of the polynomial approximation. However, it should be noted that these measurements were made under the assumption of the following bound:

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

where $q(z)$ represents the polynomial preconditioner, $\lambda_{\min}$ and $\lambda_{\max}$ are the smallest and largest eigenvalues respectively. $\delta(z)$ denotes the uniform bound for the relative approximation error on the spectral interval, given by $\delta(z) = \frac{\epsilon}{\sqrt{z}}$, with $\epsilon < \sqrt{2} - 1 \approx 0.4142$.

Using this information, the authors derived the following condition number for $A(q(A))^2$ to assess the effect of the preconditioning:

$$\kappa_{\text{pre}} \leq \frac{1 + 2\epsilon + \epsilon^2}{1 - 2\epsilon - \epsilon^2}.$$

In their numerical experiment, the authors considered a matrix $A \in \mathbb{R}^{2500\times 2500}$, representing the discretization of the Laplace operator on a square grid with 50 interior grid points in each direction. This matrix had a condition number of $\kappa(A) \approx 1054$. A Chebyshev preconditioning polynomial with $d = 32$ was applied [FRHST24]. For this specific experiment matrix, they estimated the condition number of $A(q(A))^2$ as $\kappa_{\text{pre}} \leq 1.7345$, which was verified by the experiment. The actual condition number achieved was $\kappa_{\text{pre}} = 1.5153$ for $A(q(A))^2$, slightly smaller than predicted by the bounds and approximately 700 times smaller than the condition number of A.

Further analysis of the findings presented in the paper [FRHST24] reveals that the use of polynomial preconditioning leads to significantly improved convergence for Arnoldi iterations compared to methods without preconditioning. More specifically, in the context of our application, this approach demonstrates substantial improvements in convergence. Consequently, polynomial preconditioning emerges as a promising candidate for combination with LR-deflation in our study. However, this raises the question of which polynomial preconditioner should be utilized.

A logical solution is to select a polynomial that minimizes computational effort. Additionally, it would be beneficial to choose a method that does not require extensive attention to the properties of the matrix or prior knowledge of the spectrum of A. Therefore, we favour the polynomial formed by interpolation at the (harmonic) Ritz values, as it automatically adapts to the spectrum of A.

7.1.1.3 Quadrature-based sketched FOM

The paper on randomized sketching of matrix functions [GS23] briefly explains why this method is well-suited for applications in lattice QCD problems, illustrating its effectiveness through numerical experiments in two parts.

In the first part of the experiment, a fixed Gauss-Chebyshev quadrature rule with an accuracy parameter tol $= 10^{-7}$ was used, resulting in $l = 176$ quadrature points. The maximum Krylov dimension for the experiment was $\mathrm{m}_{\max} = 300$, with a fixed sketching parameter $\mathrm{s} = 2\mathrm{m}_{\max} = 600$. Results were compared with the state-of-the-art HPC code for overlap fermion simulation [BFK+16], the quadrature-based restarted Arnoldi method. As noted in [GS23], the sketched approximations converged robustly and closely tracked the error of the best approximation. Additionally, the authors observed that convergence in the restarted method is significantly delayed, and even the largest restart length considered in the experiments led to much slower convergence than the sketching-based approach.

In the second part of the experiment, the authors measured the runtime of various methods. Results reported in [GS23] indicate that among all methods, the sketched FOM using the closed form ran the fastest. This was attributed to the need for fewer matrix-vector products, short-recurrence orthogonalization, and the absence of overhead from operations such as quadrature. Furthermore, the experimental results show that sketched FOM, the second-fastest method, saved approximately 15% of runtime and achieved higher accuracy than the quadrature-based restarted Arnoldi method. The paper concludes that quadrature-based sketching methods require slightly less than twice the time of restarted Arnoldi while also having significantly lower memory consumption, highlighting sketching-based methods as a compelling candidate for further investigation.

7.2 Combination of Deflated Quadrature-based restarted Arnoldi method and Polynomial preconditioning method

The paper [EEG11] presents an implementation of deflation in the restarted Arnoldi method, which extends the general restarted Arnoldi approach (Algorithm 8). In this approach, after each restart cycle of the Arnoldi process, a Schur decomposition of the Hessenberg matrix is used to restart the Arnoldi process with a set of targeted Ritz values.

An interesting aspect of this method is that the same approach can be incorporated into the framework of the quadrature-based restarted Arnoldi (Algorithm 9), as explained in [FGS14]. The modification of the nodal polynomials required in Algorithm 9 can be understood through Theorem 3.2 in [EEG11]. This is particularly relevant to the goals of this thesis, as we have already established that deflation acts as a catalyst to accelerate Krylov's methods.

However, we observe that the implicit quadrature-based restarted Arnoldi method experiences stagnation, at a specific relative error for various k dimensions of the Krylov subspace, indicating a slow convergence rate. Importantly, as noted in [FGS14], after an initial phase of slow convergence, the restarting method with implicit deflation exhibits the same convergence slope as the method with explicit deflation. This indicates that both methods share the same asymptotic behaviour. Here we raise the question of whether it is possible to overcome the stagnation or intermittent slow convergence observed.

In polynomial preconditioning, we noted that the preconditioner improved the condition number and had significant effects on the spectrum of the matrix. Therefore, conducting numerical experiments on the combination of implicit deflated quadrature-based restarted Arnoldi with polynomial preconditioning would be interesting. However, since polynomial preconditioned Arnoldi is computationally expensive and time-consuming, we propose adding a new parameter to control the number of polynomial preconditioned Arnoldi steps used between different cycles. This allows us to optimize the number of polynomial preconditioned Arnoldi iterations in the restarts. A framework for the implementation of the above combination is provisioned below:

Algorithm 12 Framework for Approximating $f(A)x$ using a Combination of Implicit deflated Quadrature-based restarted Arnoldi approximation and polynomial preconditioning

Given: A, b, f, m, tol, no_pre.

Compute the Polynomial preconditioned Arnoldi for A and b.
Set $f_m^{(1)} := \|b\| \|V_m^{(1)} f(H_m^{(1)}) e_1\|$.
Set $\ell := 8$ and $\tilde{\ell} := \text{round}(\sqrt{2} \cdot \ell)$.
for $k = 2, 3, \ldots$ until convergence **do**
  Compute partial Schur decomposition, $H^{(k-1)} U^{(k-1)} = U^{(k-1)} T^{(k-1)}$
  Set $Y^{(k-1)} := V^{(k-1)} U^{(k-1)}$ and reorthogonalize.
  if $k \leq no\_pre$ **then**
    Compute the Polynomial preconditioned Arnoldi.
  else
    Compute the Arnoldi decomposition $A(Y^{(k-1)} V^{(k)}) = (Y^{(k-1)} V^{(k)}) H_m^{(k)} + h_{m+1,m}^{(k)} v_{m+1}^{(k)} e_m^T$.
  end if
  Choose sets $(t_i, \omega_i)_{i=1,\ldots,\tilde{\ell}}$ and $(t_i, \omega_i)_{i=1,\ldots,\ell}$ of quadrature nodes/weights.
  Set `accurate` := `false` and `refined` := `false`.
  while `accurate` = `false` **do**
    Compute $\tilde{h}_m^{(k)} = (e_m^{(k-1)})^T f(H_m^{(k)}) e_1$ by quadrature of order $\tilde{\ell}$.
    Compute $h_m^{(k)} = (e_m^{(k-1)})^T f(H_m^{(k)}) e_1$ by quadrature of order ℓ.
    if $\|h_m^{(k)} - \tilde{h}_m^{(k)}\| < tol$ **then**
      `accurate` := `true`.
    else
      Set $\tilde{\ell} := \ell$ and $\ell := \text{round}(\sqrt{2} \cdot \tilde{\ell})$.
      Set `refined` := `true`.
    end if
  end while
  Set $f_m^{(k)} := f_m^{(k-1)} + \|b\| \|V_m^{(k)} h_m^{(k)}\|$.
  if `refined` = `false` **then**
    Set $\ell := \tilde{\ell}$ and $\tilde{\ell} := \text{round}(\ell / \sqrt{2})$.
  end if
end for

List of Figures

List of Sourcecodes

List of Tables

Glossary

Rekursion

see Rekursion

Bibliography

[AEEG08] Afanasjew, Martin; Eiermann, Michael; Ernst, Oliver G, and Güttel, Stefan. "Implementation of a restarted Krylov subspace method for the evaluation of matrix functions". In: *Linear Algebra and its applications* 429.10 (2008), pp. 2293–2314.

[AL09] Arioli, Mario and Loghin, Daniel. "Discrete Interpolation Norms with Applications". In: *SIAM Journal on Numerical Analysis* 47.4 (2009), pp. 2924–2951. DOI: 10.1137/080729360. eprint: https://doi.org/10.1137/080729360. URL: https://doi.org/10.1137/080729360.

[BN19] Balabanov, Oleg and Nouy, Anthony. "Randomized linear algebra for model reduction. Part I: Galerkin methods and error estimation". In: *Advances in Computational Mathematics* 45.5 (2019), pp. 2969–3019.

[BFLW07] Bloch, Jacques; Frommer, Andreas; Lang, Bruno, and Wettig, Tilo. "An iterative method to compute the sign function of a non-Hermitian matrix and its application to the overlap Dirac operator at nonzero chemical potential". In: *Computer Physics Communications* 177.12 (2007), pp. 933–943.

[BW06a] Bloch, Jacques and Wettig, Tilo. "Overlap Dirac operator at nonzero chemical potential and random matrix theory". In: *Physical review letters* 97.1 (2006), p. 012003.

[BW06b] Bloch, Jacques and Wettig, Tilo. "Overlap Dirac operator at nonzero chemical potential and random matrix theory". In: *Physical review letters* 97.1 (2006), p. 012003.

[Boo] Boor, C de. "Divided differences, Surv. Approx. Theory 1 (2005) 46–69". In: *arXiv preprint math/0502036* ().

[Bor00] Boriçi, Artan. "Fast methods for computing the Neuberger operator". In: *Numerical Challenges in Lattice Quantum Chromodynamics: Joint Interdisciplinary Workshop of John von Neumann Institute for Computing, Jülich, and Institute of Applied Computer Science, Wuppertal University, August 1999.* Springer. 2000, pp. 40–47.

[BFK+16] Brannick, James; Frommer, Andreas; Kahl, Karsten; Leder, Björn; Rottmann, Matthias, and Strebel, Artur. "Multigrid preconditioning for the overlap operator in lattice QCD". In: *Numerische Mathematik* 132.3 (2016), pp. 463–490.

[BHK12] Burrage, Kevin; Hale, Nicholas, and Kay, David. "An efficient implicit FEM scheme for fractional-in-space reaction-diffusion equations". In: *SIAM Journal on Scientific Computing* 34.4 (2012), A2145–A2172.

[Bye86] Byers, Ralph. "Numerical stability and instability in matrix sign function based algorithms". In: *Computational and combinatorial methods in systems theory* (1986), pp. 185–200.

[BHM97] Byers, Ralph; He, Chunyang, and Mehrmann, Volker. "The matrix sign function method and the computation of invariant subspaces". In: *SIAM Journal on Matrix Analysis and Applications* 18.3 (1997), pp. 615–632.

[Car12] Cardoso, Joao R. "Computation of the matrix pth root and its Fréchet derivative by integrals". In: *Electron. Trans. Numer. Anal* 39 (2012), pp. 414–436.

[DJ74] DePrima, Charles R and Johnson, Charles R. "The range of A- 1A in GL (n, C)". In: *Linear Algebra and its Applications* 9 (1974), pp. 209–222.

[DK89] Druskin, Vladimir L and Knizhnerman, Leonid A. "Two polynomial methods of calculating functions of symmetric matrices". In: *USSR Computational Mathematics and Mathematical Physics* 29.6 (1989), pp. 112–121.

[DK98] Druskin, Vladimir and Knizhnerman, Leonid. “Extended Krylov subspaces: approximation of the matrix square root and related functions”. In: *SIAM Journal on Matrix Analysis and Applications* 19.3 (1998), pp. 755–771.

[DFF+08] Durr, S; Fodor, Zoltàn; Frison, Julien; Hoelbling, Christian; Hoffmann, Roland; Katz, Sandor D; Krieg, Stefan; Kurth, Thorsten; Lellouch, Laurent; Lippert, Thomas, et al. “Ab initio determination of light hadron masses”. In: *Science* 322.5905 (2008), pp. 1224–1227.

[EE06] Eiermann, Michael and Ernst, Oliver G. “A restarted Krylov subspace method for the evaluation of matrix functions”. In: *SIAM Journal on Numerical Analysis* 44.6 (2006), pp. 2481–2504.

[EEG11] Eiermann, Michael; Ernst, Oliver G, and Güttel, Stefan. “Deflated restarting for matrix functions”. In: *SIAM Journal on Matrix Analysis and Applications* 32.2 (2011), pp. 621–641.

[EFL+02] Eshof, Jasper van den; Frommer, Andreas; Lippert, Th; Schilling, Klaus, and Vorst, Henk A van der. “Numerical methods for the QCDd overlap operator. I. Sign-function and error bounds”. In: *Computer physics communications* 146.2 (2002), pp. 203–224.

[FX23] Fidkowski, Lukasz and Xu, Cenke. “A No-Go Result for Implementing Chiral Symmetries by Locality-Preserving Unitaries in a Three-Dimensional Hamiltonian Lattice Model of Fermions”. In: *Physical Review Letters* 131.19 (2023), p. 196601.

[FGS14] Frommer, Andreas; Güttel, Stefan, and Schweitzer, Marcel. “Efficient and stable Arnoldi restarts for matrix functions based on quadrature”. In: *SIAM Journal on Matrix Analysis and Applications* 35.2 (2014), pp. 661–683.

[FRHST24] Frommer, Andreas; Ramirez-Hidalgo, Gustavo; Schweitzer, Marcel, and Tsolakis, Manuel. “Polynomial Preconditioning for the Action of the Matrix Square Root and Inverse Square Root”. In: *arXiv preprint arXiv:2401.06684* (2024).

[Gau91] Gautschi, Walter. “Quadrature formulae on half-infinite intervals”. In: *BIT Numerical Mathematics* 31.3 (1991), pp. 437–446.

[Güt13] Güttel, Stefan. “Rational Krylov approximation of matrix functions: Numerical methods and optimal pole selection”. In: *GAMM-Mitteilungen* 36.1 (2013), pp. 8–31.

[GK13] Güttel, Stefan and Knizhnerman, Leonid. “A black-box rational Arnoldi variant for Cauchy–Stieltjes matrix functions”. In: *BIT Numerical Mathematics* 53.3 (2013), pp. 595–616.

[GS23] Güttel, Stefan and Schweitzer, Marcel. “Randomized sketching for Krylov approximations of large-scale matrix functions”. In: *SIAM Journal on Matrix Analysis and Applications* 44.3 (2023), pp. 1073–1095.

[HK84] Hasenfratz, P. and Karsch, F. "Finite fermion density on the lattice". In: *Physics Reports* 103.1 (1984), pp. 219–223. ISSN: 0370-1573. DOI: https://doi.org/10.1016/0370-1573(84)90083-8. URL: https://www.sciencedirect.com/science/article/pii/0370157384900838.

[Hig08] Higham, J. *Functions of Matrices: Theory and Computation*. 2008.

[Hig94] Higham, Nicholas J. "The matrix sign decomposition and its relation to the polar decomposition". In: *Linear Algebra and its Applications* 212 (1994), pp. 3–20.

[HL97] Hochbruck, Marlis and Lubich, Christian. "On Krylov Subspace Approximations to the Matrix Exponential Operator". In: *SIAM Journal on Numerical Analysis* 34.5 (1997), pp. 1911–1925. DOI: 10.1137/S0036142995280572. eprint: https://doi.org/10.1137/S0036142995280572. URL: https://doi.org/10.1137/S0036142995280572.

[HO10] Hochbruck, Marlis and Ostermann, Alexander. "Exponential integrators". In: *Acta Numerica* 19 (2010), 209–286. DOI: 10.1017/S0962492910000048.

[ITS09] Ilić, M.; Turner, I. W., and Simpson, D. P. "A restarted Lanczos approximation to functions of a symmetric matrix". In: *IMA Journal of Numerical Analysis* 30.4 (June 2009), pp. 1044–1061. ISSN: 0272-4979. DOI: 10.1093/imanum/drp003. eprint: https://academic.oup.com/imajna/article-pdf/30/4/1044/2066924/drp003.pdf. URL: https://doi.org/10.1093/imanum/drp003.

[KMA+22] Karthein, Jamie M; Mroczek, Debora; Acuna, Angel Nava; Noronha-Hostler, Jacquelyn; Parotto, Paolo; Price, Damien RP, and Ratti, Claudia. "Lattice-QCD-based equations of state at finite temperature and density". In: *arXiv preprint arXiv:2211.04566* (2022).

[KL94] Kenney, Charles S and Laub, Alan J. "A hyperbolic tangent identity and the geometry of Padé sign function iterations". In: *Numerical Algorithms* 7.2 (1994), pp. 111–128.

[KL95] Kenney, Charles S and Laub, Alan J. "The matrix sign function". In: *IEEE transactions on automatic control* 40.8 (1995), pp. 1330–1348.

[KMS+83] Kogut, J; Matsuoka, H; Stone, M; Wyld, HW; Shenker, S; Shigemitsu, J, and Sinclair, DK. "Chiral symmetry restoration in baryon rich environments". In: *Nuclear Physics B* 225.1 (1983), pp. 93–122.

[MT20] Martinsson, Per-Gunnar and Tropp, Joel A. "Randomized numerical linear algebra: Foundations and algorithms". In: *Acta Numerica* 29 (2020), pp. 403–572.

[MVL03] Moler, Cleve and Van Loan, Charles. "Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later". In: *SIAM Review* 45.1 (2003), pp. 3–49. DOI: 10.1137/S00361445024180. eprint: https://doi.org/10.1137/S00361445024180. URL: https://doi.org/10.1137/S00361445024180.

[MM94] Montvay, István and Münster, Gernot. *Quantum fields on a lattice*. Cambridge University Press, 1994.

[NT22] Nagai, Yuki and Tomiya, Akio. “Extensively parallelizable chiral fermion”. In: *arXiv preprint arXiv:2204.01583* (2022).

[NT24] Nakatsukasa, Yuji and Tropp, Joel A. “Fast and accurate randomized algorithms for linear systems and eigenvalue problems”. In: *SIAM Journal on Matrix Analysis and Applications* 45.2 (2024), pp. 1183–1214.

[Neu98] Neuberger, Herbert. “A Practical implementation of the overlap Dirac operator”. In: *Physical Review Letters* 81.19 (1998), p. 4060.

[NN81] Nielsen, Holger Bech and Ninomiya, Masao. *No-go theorum for regularizing chiral fermions*. Tech. rep. Science Research Council, 1981.

[PJE+20] Pleiss, Geoff; Jankowiak, Martin; Eriksson, David; Damle, Anil, and Gardner, Jacob. “Fast matrix square roots with applications to Gaussian processes and Bayesian optimization”. In: *Advances in neural information processing systems* 33 (2020), pp. 22268–22281.

[Rob80] Roberts, John Douglas. “Linear model reduction and solution of the algebraic Riccati equation by use of the sign function”. In: *International Journal of Control* 32.4 (1980), pp. 677–687.

[Saa92] Saad, Yousef. “Analysis of some Krylov subspace approximations to the matrix exponential operator”. In: *SIAM Journal on Numerical Analysis* 29.1 (1992), pp. 209–228.

[Saa03] Saad, Yousef. *Iterative methods for sparse linear systems*. SIAM, 2003.

[SVR08] Schilders, Wilhelmus HA; Vorst, Henk A Van der, and Rommes, Joost. *Model order reduction: theory, research aspects and applications*. Vol. 13. Springer, 2008.

[Sch16] Schweitzer, Marcel. “Restarting and error estimation in polynomial and extended Krylov subspace methods for the approximation of matrix functions”. PhD thesis. University of Wuppertal, 2016.

[SS76] Singer, Burton and Spilerman, Seymour. “The representation of social processes by Markov models”. In: *American Journal of Sociology* 82.1 (1976), pp. 1–54.

[TE07] Tal-Ezer, Hillel. “On restart and error estimation for Krylov approximation of w=f(A)v”. In: *SIAM Journal on Scientific Computing* 29.6 (2007), pp. 2426–2441.

[Tso23] Tsolakis, Manuel. “Efficient Computation of the Action of Matrix Rational Functions and Laplace Transforms”. PhD thesis. Bergische Universität Wuppertal, 2023, 2023.