

Deterministic complexity analysis of Hermitian eigenproblems

Aleksandros Sobczyk
IBM Research and ETH Zurich
Zurich, Switzerland

Abstract

In this work we revisit the arithmetic and bit complexity of Hermitian eigenproblems. Recently, [BGVKS, FOCS 2020] proved that a (non-Hermitian) matrix A can be diagonalized with a randomized algorithm in $O(n^\omega \log^2(\frac{n}{\epsilon}))$ arithmetic operations, where $\omega \lesssim 2.371$ is the square matrix multiplication exponent, and [Shah, SODA 2025] significantly improved the bit complexity for the Hermitian case. Our main goal is to obtain similar deterministic complexity bounds for various Hermitian eigenproblems. In the Real RAM model, we show that a Hermitian matrix can be diagonalized deterministically in $O(n^\omega \log(n) + n^2 \text{polylog}(\frac{n}{\epsilon}))$ arithmetic operations, improving the classic deterministic $\tilde{O}(n^3)$ algorithms, and derandomizing the aforementioned state-of-the-art. The main technical step is a complete, detailed analysis of a well-known divide-and-conquer tridiagonal eigensolver of Gu and Eisenstat [GE95], when accelerated with the Fast Multipole Method, asserting that it can accurately diagonalize a symmetric tridiagonal matrix in nearly- $O(n^2)$ operations. In finite precision, we show that an algorithm by Schönhage [Sch72] to reduce a Hermitian matrix to tridiagonal form is stable in the floating point model, using $O(\log(\frac{n}{\epsilon}))$ bits of precision. This leads to a deterministic algorithm to compute all the eigenvalues of a Hermitian matrix in $O(n^\omega \mathcal{F}(\log(\frac{n}{\epsilon})) + n^2 \text{polylog}(\frac{n}{\epsilon}))$ bit operations, where $\mathcal{F}(b) \in \tilde{O}(b)$ is the bit complexity of a single floating point operation on b bits. This improves the best known $\tilde{O}(n^3)$ deterministic and $O(n^\omega \log^2(\frac{n}{\epsilon}) \mathcal{F}(\log(\frac{n}{\epsilon})))$ randomized complexities. We conclude with some other useful subroutines such as computing spectral gaps, condition numbers, and spectral projectors, and with some open problems.

Contents

1	Introduction	3
1.1	Models of computation and complexity	3
1.2	Notation	4
1.3	Methods and Contributions	4
1.3.1	Real RAM	4
1.3.2	Finite precision	7
1.4	Outline	8
2	Diagonalization of symmetric tridiagonal and arrowhead matrices in Real RAM	9
2.1	Symmetric arrowhead diagonalization	9
2.2	Tridiagonal diagonalization	10
2.3	Hermitian diagonalization	12
3	Stability of tridiagonal reduction	12
3.1	Matrix nomenclature	12
3.2	Rotations	13
3.3	Recursive bandwidth halving	13
3.4	Eigenvalues of Hermitian matrices	14
4	Further applications of stable tridiagonal reduction and Hermitian eigenvalue solver	15
4.1	Singular values and condition number	15
4.2	Definite pencil eigenvalues	17
4.3	Spectral gaps	17
4.4	Spectral projectors and invariant subspaces	19
4.5	Stable inversion and factorizations	20
5	Conclusion	20
A	Preliminaries for symmetric arrowhead diagonalization	26
A.1	Deflation	27
A.2	Reconstruction from approximate eigenvalues	29
B	Fast Arrowhead diagonalization	31
B.1	Fast Multipole Method	31
B.2	Computing the eigenvalues with bisection	34
B.3	Approximating the elements of the shaft	35
B.4	Approximating inner products with the eigenvectors	36
B.5	Proof of Theorem 2.1	41
C	Tridiagonal diagonalization	43
C.1	Omitted proofs	43
C.2	Approximating only the eigenvalues of a tridiagonal matrix	46
C.3	Application: Hermitian diagonalization	46
C.4	Singular Value Decomposition	46
D	Floating point arithmetic	47
E	Reduction to tridiagonal form - omitted proofs and definitions	48
E.1	Imported subroutines	48
E.2	Rotations	50
E.3	Bandwidth halving	52

1 Introduction

Eigenproblems arise naturally in many applications. Given a matrix A , the goal is to compute (a subset of) the eigenvalues λ and/or the eigenvectors v , which satisfy

$$Av = \lambda v.$$

The properties of the given matrix, as well as the quantities that need to be computed can vary depending on the application, giving rise to different variants of the eigenproblem. These include (i) *eigenvalue problems*, such as the approximation of eigenvalues, singular values, spectral gaps, and condition numbers, (ii) *eigenspace problems*, which refer to the approximation of eigenvectors, spectral projectors, and invariant subspaces, and (iii) *diagonalization problems*, which involve the (approximate) computation of all the eigenvalues and eigenvectors of a matrix (or pencil), i.e., a full spectral factorization and/or the Singular Value Decomposition (SVD).

In this work we focus on algorithms for *Hermitian eigenproblems*, i.e., the special case where the input matrix is Hermitian. Our motivation to dedicate the analysis to this special class is twofold. First, they arise in many fundamental applications in Machine Learning, Spectral Graph Theory, and Scientific Computing. For example, the SVD, which is ubiquitous for many applications such as low rank approximations [73, 41, 37, 22], directly reduces to a Hermitian eigenproblem. Second, the spectral theorem states that a Hermitian matrix A can always be written in a factored form $A = Q\Lambda Q^*$, where Q is unitary and Λ is diagonal with real entries. This alleviates several difficulties of the non-Hermitian case, which can lead to efficient dedicated algorithms.

Algorithms for eigenproblems have been studied for decades, some of the earliest being attributed to Jacobi [55, 45]. We refer to standard textbooks for an overview of the rich literature [33, 46, 74, 12, 77]. Some landmark works include the power method [64], the Lanczos algorithm [59], and the paramount QR algorithm [39, 40, 58], which has been recognized as one of the “top ten algorithms of the twentieth century” [35], signifying the importance of the eigenproblem in science and engineering. Given a Hermitian tridiagonal matrix T with size $n \times n$, the algorithm can compute a set of approximate eigenvalues in $O(n^2 \log(\frac{n}{\epsilon}))$ arithmetic operations, based on the seminal analyses of Wilkinson [88] and Dekker and Traub [27]. A set of approximate eigenvectors can be also returned in $O(n^3 \log(\frac{n}{\epsilon}))$ operations. In conjunction with the classic unitary similarity transformation algorithm of Householder [54], the shifted-QR algorithm has heavily lifted the computational burden of solving eigenvalue problems for decades, both in theory and in practice. A detailed bit complexity analysis was provided recently in [8, 7, 9].

Despite the daunting literature, several details regarding the computational complexity of many algorithms remain unclear. It is well-known, for example, that the cubic arithmetic complexity to compute the eigenvalues of a dense matrix is not optimal: a classic work by Pan and Chen [72] showed that the eigenvalues can be approximated in $O(n^\omega)$ arithmetic operations, albeit without detailing how to also compute eigenvectors, or to fully diagonalize a matrix. Here $\omega \geq 2$ is the *matrix multiplication exponent*, i.e. the smallest number such that two $n \times n$ matrices can be multiplied in $O(n^{\omega+\eta})$ arithmetic operations, for any $\eta > 0$. The current best known upper bound is $\omega < 2.371339$ [1], and we will write n^ω instead of $n^{\omega+\eta}$ hereafter for simplicity. Recently, Banks, Garza-Vargas, Kulkarni, and Srivastava [6] described a numerically stable randomized algorithm to compute a provably accurate diagonalization, in $\tilde{O}(n^\omega)$ operations, improving the previous best-known bounds, specifically, $O(n^3)$ (Hermitian) and $O(n^{10})$ (non-Hermitian) [3]. [82] further improved the analysis for the Hermitian case, and several works have studied extensions and related applications [30, 31, 78, 83]. To date, we are not aware of any *deterministic* algorithm that achieves the same arithmetic (or bit) complexity with provable approximation guarantees, even for the Hermitian case. In this work, we proceed step-by-step and analyze several variants of the aforementioned eigenvalue, eigenspace, and diagonalization problems, in different *models of computation*, and report complexity upper bounds with provable approximation guarantees.

1.1 Models of computation and complexity

From the Abel-Ruffini theorem it is known that the eigenvalues and/or the eigenvectors of matrices with size larger than $n = 5$ can not be computed exactly. Even in exact arithmetic, they can only be approximated. Before analyzing algorithms, we first need to clarify what are the quantities of interest, to define how accuracy is measured, and what is the underlying model of computation. The main two models that we use to analyze algorithms are the Real RAM and the Floating Point model, described below.

Exact real arithmetic (Real RAM): For the Real RAM model we follow the definition of [38]. The machine has a memory (RAM) and registers that store real numbers in infinite precision. Moving real numbers between the memory and the registers takes constant time. A processor can execute arithmetic operations $\{+, -, \times, /, \sqrt{\cdot}, >\}$ on the real numbers stored in the registers exactly, without any errors, in constant time. Other functions such as $\log(x)$, $\exp(x)$, and trigonometric functions, are not explicitly available, but they can be efficiently approximated up to some additive error, e.g. with a truncated polynomial expansion, using a polylogarithmic number of basic arithmetic operations. Bit-wise operations on real numbers are forbidden, since this can give the machine unreasonable computing power [50, 80, 38].

Floating point: In this model, a real number $\alpha \in \mathbb{R}$ is rounded to a floating point number $\mathbf{fl}(\alpha) = s \times 2^{e-t} \times m$, where $s = \pm 1$, e is the exponent, t is the bias, and m is the significand. Floating point arithmetic operations also introduce rounding errors, i.e., for two floating point numbers α and β , each operation $\odot \in \{+, -, \times, /\}$ satisfies:

$$\mathbf{fl}(\alpha \odot \beta) = (1 + \theta)(\alpha \odot \beta), \quad \text{and also} \quad \mathbf{fl}(\sqrt{a}) = (1 + \theta)\sqrt{a}, \quad (1)$$

where $|\theta| \leq \mathbf{u}$, and \mathbf{u} is the machine precision. Assuming a total of b bits for each number, every floating point operation costs $\mathcal{F}(b)$ bit operations, where typically $\mathcal{F}(b) \in O(b^2)$, or even $\tilde{O}(b)$, with more advanced algorithms [81, 42, 51]. More details can be found in Appendix D.

In Section 3.4 we will also use as a subroutine an algorithm from [13, 15], which was originally analyzed in Boolean RAM model. We describe in detail how to use it in the corresponding section.

Arithmetic and boolean complexity Given a model of computation, the *arithmetic complexity* of an algorithm is quantified in terms of the arithmetic operations executed. The *boolean* (or *bit*) *complexity*, accounts for the total number of boolean operations (i.e. boolean gates with maximum fan-in two).

1.2 Notation

Matrices and vectors are denoted with bold capital and small letters, respectively. $\|\mathbf{A}\|$ is the spectral norm of \mathbf{A} , $\|\mathbf{A}\|_F$ its Frobenius norm, $\kappa(\mathbf{A}) = \|\mathbf{A}\| \|\mathbf{A}^\dagger\|$ is the condition number, and $\Lambda(\mathbf{A})$ its spectrum. For the complexity analysis we denote the geometric series $S_x(m) = \sum_{l=1}^m (2^{x-2})^l$. As already mentioned, for simplicity, the complexity of multiplying two $n \times n$ matrices will be denoted as $O(n^\omega)$, instead of $O(n^{\omega+\eta})$, for arbitrarily small $\eta > 0$. We also use the standard notation $\omega(a, b, c)$ for the complexity of multiplying two rectangular matrices with sizes $n^a \times n^b$ and $n^b \times n^c$ in time $O(n^{\omega(a,b,c)})$, and therefore $\omega := \omega(1, 1, 1)$. For example, for $a = c = 1$ and $b = 2$, the best known bound for $\omega(1, 2, 1) \approx 3.250035$ [1], which is slightly better than naively performing n square multiplications in $O(n \cdot n^\omega) \lesssim O(n^{3.371339})$. $\mathcal{F}(b)$ denotes the bit complexity of a single floating point operation on b bits.

1.3 Methods and Contributions

1.3.1 Real RAM

We start with the analysis in exact arithmetic, which is the simplest model to analyze numerical algorithms. The goal is to obtain end-to-end upper bounds for the arithmetic complexity of approximately diagonalizing symmetric tridiagonal matrices and, ultimately, dense Hermitian matrices, as well as to approximate the SVD. To measure the accuracy, we follow the notion of *backward-stability* (or *backward-approximation*) for Hermitian diagonalization from [67]. Formally, the following problems are considered.

Problem 1.1. *Backward-approximate diagonalization problems in exact arithmetic.*

- (i) **Symmetric arrowhead/tridiagonal diagonalization:** Given a symmetric arrowhead or tridiagonal matrix \mathbf{A} with size $n \times n$, $\|\mathbf{A}\| \leq 1$, and accuracy $\epsilon \in (0, 1)$, compute a diagonal matrix $\tilde{\Lambda}$ and a matrix $\tilde{\mathbf{U}}$, such that $\tilde{\mathbf{U}} = \mathbf{U} + \mathbf{E}_{\mathbf{U}}$, where \mathbf{U} is orthogonal and $\|\mathbf{E}_{\mathbf{U}}\| \leq \epsilon$, and $\left\| \mathbf{A} - \mathbf{U} \tilde{\Lambda} \mathbf{U}^\top \right\| \leq \epsilon$.

- (ii) **Hermitian diagonalization:** Given a Hermitian matrix \mathbf{A} with size $n \times n$, $\|\mathbf{A}\| \leq 1$, and accuracy $\epsilon \in (0, 1)$, compute a diagonal matrix $\tilde{\Lambda}$ and a matrix $\tilde{\mathbf{U}}$, such that $\tilde{\mathbf{U}} = \mathbf{U} + \mathbf{E}_U$, where \mathbf{U} is unitary and $\|\mathbf{E}_U\| \leq \epsilon$, and $\|\mathbf{A} - \tilde{\mathbf{U}}\tilde{\Lambda}\tilde{\mathbf{U}}^*\| \leq \epsilon$.
- (iii) **SVD:** Given a matrix $\mathbf{A} \in \mathbb{C}^{m \times n}$, $m = n^k$, $k \geq 1$, $\|\mathbf{A}\| \leq 1$, and accuracy $\epsilon \in (0, 1)$ compute a diagonal matrix $\tilde{\Sigma}$, an $m \times n$ matrix $\tilde{\mathbf{U}} = \mathbf{U} + \mathbf{E}_U$, and an $n \times n$ matrix $\tilde{\mathbf{V}} = \mathbf{V} + \mathbf{E}_V$, such that $\mathbf{U}^*\mathbf{U} = \mathbf{V}^*\mathbf{V} = \mathbf{I}$, $\|\mathbf{E}_{\{\mathbf{U}, \mathbf{V}\}}\| \leq \epsilon$, and $\|\mathbf{A} - \tilde{\mathbf{U}}\tilde{\Sigma}\tilde{\mathbf{V}}^*\| \leq \epsilon$.

To obtain rigorous complexity guarantees for these problems, with respect to all the involved parameters, we start from Problem 1.1-(i), namely, diagonalization of symmetric tridiagonal matrices. To that end, we revisit the so-called fast tridiagonal eigensolvers, which aim to reduce the complexity from cubic to $\tilde{O}(n^2)$ operations. Many such algorithms have been studied in the literature [36, 34, 14, 13, 43, 87, 70, 84, 10], most of which are based on the divide-and-conquer (DC) strategy of Cuppen [24]. The algorithms in all of the aforementioned works have been rigorously analyzed, however, explicit complexity bounds in terms of strictly solving Problem 1.1-(i) are not detailed. We resolve this by providing an end-to-end complexity analysis of the algorithm of Gu and Eisenstat [48]. In their original work, the authors outlined how to accelerate several parts of the algorithm with the Fast Multipole Method (FMM) [75], which could eventually lead to a final complexity of $\tilde{O}(n^2)$. However, the actual analysis of this approach and the FMM details were not provided. [65] further extended the analysis in floating point, but it also relies on a numerically stable FMM implementation, which is not detailed. In this work, we use the elegant FMM analysis of [47, 60, 21], which is particularly suited for the problems considered. It is detailed in the following Proposition 1.1.

Proposition 1.1 (FMM). *There exists an algorithm, which we refer to as (ϵ, n) -approximate FMM (or (ϵ, n) -FMM, for short), which takes as input:*

- a kernel function $k(x) \in \{\log(|x|), \frac{1}{x}, \frac{1}{x^2}\}$,
- $2n + m$ real numbers: $\{x_1, \dots, x_m\} \cup \{c_1, \dots, c_n\} \cup \{y_1, \dots, y_n\}$, and a constant C , such that $m \leq n$ and for all $i \in [m]$, $j \in [n]$ it holds that

$$|x_i|, |c_j|, |y_j| < C \quad \text{and} \quad |x_i - y_j| \geq \Omega(\text{poly}(\frac{\epsilon}{n})).$$

It returns m values $\tilde{f}(x_1), \dots, \tilde{f}(x_m)$ such that $|\tilde{f}(x_i) - f(x_i)| \leq \epsilon$, for all $i \in [m]$, where $f(x) = \sum_{j=1}^n c_j k(x_i - y_j)$, in a total of $O\left(n \log^\xi\left(\frac{n}{\epsilon}\right)\right)$ arithmetic operations, where $\xi \geq 1$ is a small constant that is independent of ϵ, n .

Proof. The result follows from the analysis of [47, 60, 21]. A detailed description can be found in the restatement, Proposition B.1, in Appendix B.1. \square

By taking advantage of the FMM, the analysis from Section 2 and the supporting Appendices A and B.1 leads to the following Theorem 1.1, whose proof can be found in Section 2.2.

Theorem 1.1. *Given an unreduced symmetric tridiagonal matrix \mathbf{T} with size $n \times n$, $\|\mathbf{T}\| \leq 1$, an accuracy $\epsilon \in (0, 1/2)$, and an (ϵ, n) -FMM implementation as in Proposition 1.1, the recursive algorithm of [48] (Algorithm 1), returns an approximately orthogonal matrix $\tilde{\mathbf{U}}$ and a diagonal matrix $\tilde{\Lambda}$ such that*

$$\|\mathbf{T} - \tilde{\mathbf{U}}\tilde{\Lambda}\tilde{\mathbf{U}}^\top\| \leq \epsilon, \quad \|\tilde{\mathbf{U}}^\top\tilde{\mathbf{U}} - \mathbf{I}\| \leq \epsilon/n^2,$$

or, stated alternatively,

$$\tilde{\mathbf{U}} = \mathbf{U} + \mathbf{E}_U, \quad \mathbf{U}^\top\mathbf{U} = \mathbf{I}, \quad \|\mathbf{E}_U\| \leq \epsilon/n^2, \quad \|\mathbf{T} - \mathbf{U}\tilde{\Lambda}\mathbf{U}^\top\| \leq \epsilon,$$

using a total of $O\left(n^2 \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$ arithmetic operations and comparisons, where $\xi \geq 1$ is a small constant that depends on the specific FMM implementation and it is independent of ϵ, n .

Table 1: Comparison of algorithms for the Problems 1.1 (i)-(iii) in the Real RAM model. Here $\xi \geq 1$ is a small constant which depends on the FMM implementation (see Prop. 1.1). The algorithms marked with **(R)** are randomized and succeed with high probability (at least $1 - 1/\text{poly}(n)$).

	Arithmetic Complexity	Comment
Prob. 1.1-(i) Arrowhead/Trid. diagonalization		
Refs. [24, 69, 48]	$O(n^3) + \tilde{O}(n^2)$	Conjectured $\tilde{O}(n^2)$
Theorem 1.1	$O\left(n^2 \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$	Req. FMM (Prop. 1.1)
Prob. 1.1-(ii) Hermitian diagonalization		
Refs. [6, 82] (R)	$O\left(n^\omega \log^2\left(\frac{n}{\epsilon}\right)\right)$	-
Ref. [11] (R)	$\tilde{O}\left(n^{\omega+1}\right)$	Conjectured $\tilde{O}(n^\omega)$
Refs. [27, 88, 53]	$O\left(n^3 \log\left(\frac{n}{\epsilon}\right)\right)$	Shifted-QR
Ref. [67]	$\tilde{O}\left(n^3\right)$	Req. separated spectrum
Ref. [72]	$O\left(n^\omega + n \text{polylog}\left(\frac{n}{\epsilon}\right)\right)$	Only eigenvalues
Corollary 2.1	$O\left(n^\omega \log(n) + n^2 \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$	Req. FMM (Prop. 1.1)
Prob. 1.1-(iii), SVD		
Shifted-QR on A^*A	$O\left(n^{\omega(1,k,1)} + n^3 \log\left(\frac{n}{\epsilon}\right)\right)$	Partial error analysis
Refs. [6, 82, 57] (R)	$O\left(n^{\omega(1,k,1)} + n^\omega \log^2\left(\frac{n}{\epsilon}\right)\right)$	Partial error analysis
Theorem 1.2	$O\left(n^{\omega(1,k,1)} + n^\omega \log(n) + n^2 \text{polylog}\left(\frac{n\kappa(A)}{\epsilon}\right)\right)$	Req. FMM (Prop. 1.1)

This result has a direct application to the dense Hermitian diagonalization problem. The best-known complexities of deterministic algorithms, with end-to-end analysis, are at least cubic. For example, the aforementioned shifted-QR algorithm requires $O(n^3 \log(n/\epsilon))$ arithmetic operations, even for tridiagonal matrices. The cubic barrier originates from the accumulation of the elementary rotation matrices to form the final eigenvector matrix. The so-called *spectral divide-and-conquer* methods (see e.g. [33, 67]) have also at least cubic complexities in the deterministic case. The two main difficulties in their analysis are the basis computation of spectral projectors, for which the best-known deterministic complexity bounds are $\Omega(n^3)$, e.g. via Strong Rank-Revealing QR (RRQR) factorization [49], and the choice of suitable splitting points, which relies on the existence of spectral gaps.

Randomness can help to overcome certain difficulties in the analysis. [28] analyzed a numerically stable Randomized URV decomposition, which can be used to replace RRQR for basis computations in spectral DC algorithms. A highly parallelizable Hermitian diagonalization algorithm with end-to-end analysis was proposed in [11]. While the reported sequential arithmetic complexity is $O(n^{\omega+1})$, the authors conjectured that it can be further reduced to $\tilde{O}(n^\omega)$. The first end-to-end $\tilde{O}(n^\omega)$ complexity upper bound was achieved in [6]. One of the main techniques was to use random perturbations to ensure that the pseudospectrum is well-separated, which helps to find splitting points in spectral DC. [82] further improved the analysis for Hermitian matrices.

Theorem 1.1 can be directly used to obtain a simple and deterministic solution for the Hermitian diagonalization problem. Specifically, Corollary 2.1 states that the problem can be solved in $O(n^\omega \log(n) + n^2 \text{polylog}(n/\epsilon))$ arithmetic operations, which is both faster and fully deterministic. This is achieved by combining Theorem 1.1 with the (rather overlooked) algorithm of Schönhage [79], who proved that a Hermitian matrix can be reduced to tridiagonal form with unitary similarity transformations in $O(n^\omega \log(n)c(\omega))$ arithmetic operations, where $c(\omega) = O(1)$ if ω is treated as a fixed constant larger than 2, while $c(\omega) = O(\log(n))$ if it turns out that $\omega = 2$.

As a consequence, Theorem 1.2 reports similar deterministic complexity results for the SVD. The SVD is a fundamental computational kernel in many applications, such as Principal Component Analysis [56], and it is also widely used as a subroutine for many other advanced algorithms, e.g. [73, 41, 37, 18, 20, 23, 19, 22], to name a few. A straightforward algorithm to compute it is to first form the Gramian matrix A^*A , and then diagonalize it. Other

classic SVD algorithms, such as the widely adopted Golub-Kahan bidiagonalization and its variants [44], or polar decomposition-based methods [67, 66], avoid the computation of A^*A for numerical stability reasons, but they also rely on a diagonalization algorithm as a subroutine. [57] elaborated on a matrix multiplication time SVD algorithm, by using [6] to diagonalize $A^\top A$, albeit without fully completing the backward stability analysis. The following Theorem 1.2, which is proved in Appendix C.4, states our main result.

Theorem 1.2 (SVD). *Let $A \in \mathbb{C}^{m \times n}$, $m = n^k$, $k \geq 1$. Assume that $1/n^c \leq \|A\| \leq 1$, for some constant c . Given accuracy $\epsilon \in (0, 1/2)$ and an (ϵ, n) -FMM (see Prop. 1.1), we can compute three matrices $\tilde{U} \in \mathbb{C}^{m \times n}$, $\tilde{\Sigma} \in \mathbb{R}^{n \times n}$, $\tilde{V} \in \mathbb{C}^{n \times n}$ such that $\tilde{\Sigma}$ is diagonal with positive diagonal elements and*

$$A = \tilde{U}\tilde{\Sigma}\tilde{V}^*, \quad \|\tilde{U}^*\tilde{U} - I\| \leq \epsilon, \quad \|\tilde{V}^*\tilde{V} - I\| \leq \epsilon/(\kappa(A)n^2) \ll \epsilon,$$

or, stated alternatively,

$$\begin{aligned} \|A - U\tilde{\Sigma}V^*\| &\leq \epsilon, \quad \tilde{U} = U + E_U, \quad \|E_U\| \leq \epsilon, \quad U^*U = I, \\ \tilde{V} &= V + E_V, \quad \|E_V\| \leq \epsilon/n^2, \quad V^*V = I. \end{aligned}$$

The algorithm requires a total of at most $O\left(n^{\omega(1,k,1)} + n^\omega \log(n) + n^2 \text{polylog}\left(\frac{n\kappa(A)}{\epsilon}\right)\right)$ arithmetic operations, where $\kappa(A) = \|A\|\|A^\dagger\|$.

To summarize this section, in Table 2 we list all the deterministic arithmetic complexities achieved in the Real RAM model, for all the aforementioned problems, and compare with some important existing algorithms.

1.3.2 Finite precision

Similar deterministic complexity upper bounds are obtained for several problems in finite precision. In particular, we study the following problems, for which we seek to bound the boolean complexity, i.e., the total number of bit operations.

Problem 1.2. *Main problems in finite precision.*

- (i) **Tridiagonal reduction:** *Given a Hermitian matrix A with floating point elements, reduce A to tridiagonal form using (approximate) unitary similarity transformations. In particular, return a tridiagonal matrix \tilde{T} , and (optionally) an approximately unitary matrix \tilde{Q} , such that*

$$\|\tilde{Q}\tilde{Q}^* - I\| \leq \epsilon, \quad \text{and} \quad \|A - \tilde{Q}\tilde{T}\tilde{Q}^*\| \leq \epsilon \|A\|,$$

- (ii) **Hermitian eigenvalues:** *Given a Hermitian matrix A , $\|A\| \leq 1$, and accuracy $\epsilon \in (0, 1)$, compute a set of approximate eigenvalues $\tilde{\lambda}_i$ such that $|\tilde{\lambda}_i - \lambda_i(A)| \leq \epsilon$.*

Regarding deterministic algorithms, with end-to-end-analysis, the standard approach is to first reduce the Hermitian matrix to tridiagonal form with Householder transformations [54], which can be done stably in $O(n^3)$ arithmetic operations using $O(\log(n/\epsilon))$ bits of precision; see e.g. [52]. Thereafter, there is a plethora of algorithms (e.g. the ones mentioned in the previous section) for the eigenvalues of the tridiagonal matrix, with varying complexities and stability properties. However, the total boolean complexity cannot be lower than $\Omega(n^3 \mathcal{F}(\log(\frac{n}{\epsilon})))$ due to the Householder reduction step. Other well-known deterministic and numerically stable algorithms in the literature also require at least n^3 arithmetic operations to compute all the eigenvalues [71, 32, 67], and at least $\text{polylog}(n, 1/\epsilon)$ bits of precision in a floating point machine. The arithmetic complexity of the algorithm of [72] scales as $O(n^\omega)$ with respect to the matrix size n , but the boolean complexity can increase up to $O(n^{\omega+1})$ in rational arithmetic. [61] described a randomized algorithm to compute only the largest eigenvalue in nearly $O(n^\omega)$ bit complexity. The fastest algorithm to compute all the eigenvalues of a Hermitian matrix is [82], which requires $O(n^\omega \text{polylog}(n/\epsilon))$ boolean operations and succeeds with high probability.

Table 2: Boolean complexity for Problems 1.2, for matrix size $n \times n$ and accuracy $\epsilon \in (0, 1)$.

	Boolean Complexity	Comment
Prob. 1.2-(i)		
Tridiag. Reduction		
Refs. [54, 52]	$O\left(n^3 \mathcal{F}\left(\log\left(\frac{n}{\epsilon}\right)\right)\right)$	Standard Householder reduction
Theorem 3.1	$O\left(n^\omega \log(n) \mathcal{F}\left(\log\left(\frac{n}{\epsilon}\right)\right)\right)$	[79] with stable fast QR [28]
Prob. 1.2-(ii)		
Herm. Eigenvalues		
Ref. [82]	$O\left(n^\omega \log^2\left(\frac{n}{\epsilon}\right) \mathcal{F}\left(\log\left(\frac{n}{\epsilon}\right)\right)\right)$	Randomized, $\Pr[\text{success}] \geq 1 - \frac{1}{n}$
Theorem 3.3	$O\left(n^\omega \mathcal{F}\left(\log\left(\frac{n}{\epsilon}\right)\right) + n^2 \text{polylog}\left(\frac{n}{\epsilon}\right)\right)$	Deterministic, Thm. 3.1 + [15]

Randomized eigenvalue algorithms have also been studied in the sketching/streaming setting [2, 68, 86]. The (optimal) algorithm of [86] has not been analyzed in finite-precision, but, due to its simplicity, it should be straightforward to achieve. The algorithm approximates all the eigenvalues of a Hermitian matrix A up to additive error $\epsilon \|A\|_F$. However, to reduce the error to a spectral-norm bound $\epsilon \|A\|$, the algorithm internally needs to diagonalize a matrix with size $\Omega(n)$, and therefore it does not provide any improvement against any other Hermitian eigenvalue solver. Nevertheless, our main results can be directly applied as the main eigenvalue subroutine of this algorithm, and to help analyze its bit complexity.

To improve the aforementioned Hermitian eigenvalue algorithms, we first prove in Theorem 3.1 it is proved that Schönhage’s algorithm is numerically stable in the floating point model of computation. Thereafter, we carefully combine it with the algorithms of [13, 14, 15] which provably and deterministically approximate all the eigenvalues of a symmetric tridiagonal matrix in $\tilde{O}(n^2)$ boolean operations. The latter algorithm is analyzed in the Boolean RAM model, therefore, in order to use it we need to convert the floating point elements of the tridiagonal matrix that is returned by Theorem 3.1 to integers, which can be done efficiently under reasonable assumptions on the initial number of bits used to represent the floating point numbers. This is described in detail in the proof of our main Theorem 3.3. Our result derandomizes and slightly improves the final bit complexity of the algorithm of [82], which has the currently best known bit complexity for this problem. Table 2 summarizes this discussion.

As a direct consequence of Theorem 3.3, we also provide the analysis for several other useful subroutines related to eigenvalue/eigenvector computations, including:

- (i) **Singular values and condition number:** In Proposition 4.1 we describe how to obtain relative error approximations of singular values. In Corollary 4.1 we show how to compute the condition number.
- (ii) **Definite pencil eigenvalues:** In Corollary 4.2 we demonstrate how to extend Theorem 3.3 to compute the eigenvalues of Hermitian-definite pencils.
- (iii) **Spectral gaps:** In Corollary 4.4 we show how to compute the spectral gap and the midpoint between any two eigenvalues of a Hermitian-definite pencil. Our algorithm is deterministic and it requires significantly less bits of precision than the algorithm of [83], who described a randomized algorithm for this problem that is slightly faster than applying [6] as a black-box, but it only computes a single spectral gap.
- (iv) **Spectral projector:** Corollary 4.5 details how to compute spectral projectors on invariant subspaces of Hermitian-definite pencils, which are important for many applications.

1.4 Outline

The paper is organized as follows. In Section 2 we analyze the algorithm of [48] when implemented with the FMM and its applications (see also Appendices A, B.1, and C). In Section 3 it is proved that Schönhage’s algorithm is numerically stable in floating point, and it is used as a preprocessing step to compute the eigenvalues of Hermitian matrices. For the proof we use the technical lemmas that are proved in the supporting Appendix E. In Section 4

we mention some direct applications to compute singular values, pencil eigenvalues, spectral gaps, and spectral projectors. We finally conclude and state some open problems in Section 5.

Acknowledgements

I am grateful to Efstratios Gallopoulos, Daniel Kressner, and David Woodruff for helpful discussions.

2 Diagonalization of symmetric tridiagonal and arrowhead matrices in Real RAM

Our main target is to compute the eigenvalues and the eigenvectors of tridiagonal symmetric matrices in nearly linear time. To derive the desired result, we provide an analysis the divide-and-conquer algorithm of Gu and Eisenstat [48], when implemented with the FMM from Proposition 1.1.

The algorithm of [48] first “divides” the problem by partitioning the (unreduced) tridiagonal matrix T as follows:

$$T = \begin{pmatrix} T_1 & \beta_{k+1}\mathbf{e}_k & \\ \beta_{k+1}\mathbf{e}_k^\top & \alpha_{k+1} & \beta_{k+2}\mathbf{e}_1^\top \\ & \beta_{k+2}\mathbf{e}_1 & T_2 \end{pmatrix}.$$

If one has access to the spectral decomposition of T_1 and T_2 , i.e. $T_1 = Q_1 D_1 Q_1^\top$ and $T_2 = Q_2 D_2 Q_2^\top$, then T can be factorized as

$$\begin{pmatrix} & Q_1 \\ 1 & \\ & Q_2 \end{pmatrix} \begin{pmatrix} \alpha_{k+1} & \beta_{k+1}\mathbf{l}_1^\top & \beta_{k+2}\mathbf{f}_2^\top \\ \beta_{k+1}\mathbf{l}_1 & D_1 & \\ \beta_{k+2}\mathbf{f}_2 & & D_2 \end{pmatrix} \begin{pmatrix} Q_1^\top & 1 \\ & Q_2^\top \end{pmatrix} = QHQ^\top, \quad (2)$$

where \mathbf{l}_1^\top is the last row of Q_1 and \mathbf{f}_2^\top is the first row of Q_2 , and H has the so-called *arrowhead* structure. Thus, given this form, to compute the spectral decomposition of T , it suffices to diagonalize H . One can then apply recursively the algorithm to compute the spectral decompositions of T_1 and T_2 , and, finally, at the “conquering stage,” combine the solutions with the eigendecomposition of H .

For the individual steps to be computed efficiently, we will need to use the FMM. Specifically, we will need to use it to evaluate the functions that are listed in Equations (12)-(16), where the evaluation points will also satisfy certain criteria that are detailed in Lemma A.1. We ensure that these criteria are met by using a deflation pre-processing step (see also Appendix A.1), which allows us to take advantage of the FMM, specifically, Proposition 1.1.

2.1 Symmetric arrowhead diagonalization

The first step is to provide an end-to-end complexity analysis for the arrowhead diagonalization algorithm of [48], when implemented with the FMM. We start with an $n \times n$ arrowhead matrix of the form

$$H = \begin{pmatrix} \alpha & \mathbf{z}^\top \\ \mathbf{z} & D \end{pmatrix}, \quad (3)$$

where D is a diagonal matrix, \mathbf{z} is a vector of size $n - 1$ and α is a scalar. Without loss of generality we assume that $\|H\| \leq 1$. The main result is stated in Theorem 2.1.

In order to prove Theorem 2.1, we expand in detail the following methodology which is outlined in [48], by proving several technical lemmas in Appendices A and B.1 that leverage the FMM.

1. Deflation: The matrix H is preprocessed to ensure that it satisfies the following: $d_{i+1} - d_i \geq \tau$, and $|\mathbf{z}_i| \geq \tau$, where $\tau \in (0, 1)$. This assumption implies several useful properties, described in Lemma A.1, and it allows us to efficiently utilize the FMM in the subsequent steps. The deflation procedure is described in Proposition A.1.

2. Eigenvalues: The eigenvalues of the deflated matrix can be conveniently approximated as the roots of the corresponding secular equation (Eq. (6)): $f(\lambda) = \lambda - \alpha + \sum_{j=2}^n \frac{z_j^2}{d_j - \lambda}$. An FMM-based root finder is detailed in Lemma B.1.
3. From Lemma A.2, the approximate eigenvalues returned by Lemma B.1 are the exact eigenvalues of another arrowhead matrix $\widehat{\mathbf{H}} = \begin{pmatrix} \widehat{\alpha} & \widehat{\mathbf{z}}^\top \\ \widehat{\mathbf{z}} & \mathbf{D} \end{pmatrix}$. There is an analytical expression for the elements of $\widehat{\mathbf{H}}$ (see also [17, 48]). Lemma B.2 describes how to compute those elements with the FMM.
4. Given the exact eigenvalues and the approximate elements of the matrix $\widehat{\mathbf{H}}$, we can focus on its eigenvectors. In particular, in Lemma B.3 it is shown how to use the FMM to approximate the inner products between the eigenvectors of $\widehat{\mathbf{H}}$ and some arbitrary unit vector \mathbf{b} . Computing such inner products with all the columns of the identity, we obtain the final approximate eigenvector matrix of \mathbf{H} and, ultimately, an approximate diagonalization of \mathbf{H} .

Remark 2.1. We note that, if we are only interested in a full diagonalization, Lemma B.2 is redundant, i.e., we can naively compute the elements exactly without the FMM with the same complexity. However, it is useful if we need to compute only a few matrix-vector products with the eigenvector matrix. Lemma B.3, details how to approximate such matrix-vector products efficiently with the FMM.

Theorem 2.1. Given a symmetric arrowhead matrix $\mathbf{H} \in \mathbb{R}^{n \times n}$ as in Eq. (3), with $\|\mathbf{H}\| \leq 1$, an accuracy parameter $\epsilon \in (0, 1)$, a matrix \mathbf{B} with r columns $\mathbf{B}_i, i \in [r]$, where $\|\mathbf{B}_i\| \leq 1$, and an (ϵ, n) -FMM implementation (see Prop. 1.1), we can compute a diagonal matrix $\widetilde{\Lambda}$, and a matrix $\widetilde{\mathbf{Q}}_{\mathbf{B}}$, such that

$$\left\| \mathbf{H} - \mathbf{Q} \widetilde{\Lambda} \mathbf{Q}^\top \right\| \leq \epsilon, \quad \left| \left(\mathbf{Q}^\top \mathbf{B} - \widetilde{\mathbf{Q}}_{\mathbf{B}} \right)_{i,j} \right| \leq \epsilon/n^2,$$

where $\mathbf{Q} \in \mathbb{R}^{n \times n}$ is (exactly) orthogonal, in $O\left(nr \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$ arithmetic operations and comparisons, where $\xi \geq 1$ is a small constant that depends on the specific FMM implementation and it is independent of ϵ, n .

Alternatively, if only want to compute a set of approximate values $\widetilde{\lambda}_1, \dots, \widetilde{\lambda}_n$, such that $|\lambda_i(\mathbf{H}) - \widetilde{\lambda}_i| \leq \epsilon$, the complexity reduces to $O\left(n \log\left(\frac{1}{\epsilon}\right) \log^\xi\left(\frac{n}{\epsilon}\right)\right)$ arithmetic operations.

Proof. The full proof is provided in the restatement in Theorem B.1 in the Appendix. Briefly, we start with a deflation step in order to ensure that the properties of Lemma A.1 are satisfied. Next, we compute the eigenvalues using the bisection algorithm of Lemma B.1. Given the eigenvalues, we can use Lemmas B.2 to compute the elements of the “reconstructed” matrix $\widehat{\mathbf{H}}$, and finally use Lemma B.3 to approximate inner products with the eigenvectors of $\widehat{\mathbf{H}}$, which give the final approximate eigenvectors of \mathbf{H} . \square

2.2 Tridiagonal diagonalization

Given the analysis for arrowhead diagonalization, we can now proceed to tridiagonal matrices. The next lemma bounds the error of the reduction to arrowhead form when the spectral factorizations of the matrices \mathbf{T}_1 and \mathbf{T}_2 in Equation (2) are approximate rather than exact. This will be used as an inductive step for the final algorithm.

Lemma 2.1. Let $\epsilon \in (0, 1/2)$ be a given accuracy parameter and $\mathbf{T} = \begin{pmatrix} \mathbf{T}_1 & \beta_{k+1} \mathbf{e}_k & \\ \beta_{k+1} \mathbf{e}_k^\top & \alpha_{k+1} & \beta_{k+2} \mathbf{e}_1^\top \\ & \beta_{k+2} \mathbf{e}_1 & \mathbf{T}_2 \end{pmatrix}$ be a tridiagonal matrix with size $n \geq 3$ and $\|\mathbf{T}\| \leq 1$, where $\mathbf{T}_1 = \mathbf{U}_1 \mathbf{D}_1 \mathbf{U}_1^\top$ and $\mathbf{T}_2 = \mathbf{U}_2 \mathbf{D}_2 \mathbf{U}_2^\top$ be the exact spectral factorizations of \mathbf{T}_1 and \mathbf{T}_2 . Let $\widetilde{\mathbf{U}}_1, \widetilde{\mathbf{D}}_1, \widetilde{\mathbf{U}}_2, \widetilde{\mathbf{D}}_2$ be approximate spectral factorizations of $\mathbf{T}_1, \mathbf{T}_2$. If these factors satisfy

$$\left\| \mathbf{T}_{\{1,2\}} - \widetilde{\mathbf{U}}_{\{1,2\}} \widetilde{\mathbf{D}}_{\{1,2\}} \widetilde{\mathbf{U}}_{\{1,2\}}^\top \right\| \leq \epsilon_1, \quad \left\| \widetilde{\mathbf{U}}_{\{1,2\}} \widetilde{\mathbf{U}}_{\{1,2\}}^\top - \mathbf{I} \right\| \leq \epsilon_1/n,$$

for some $\epsilon_1 \in (0, 1/2)$, where $\tilde{\mathbf{D}}_{\{1,2\}}$ are both diagonal, then, assuming an (ϵ, n) -FMM implementation as in Prop. 1.1, we can compute a diagonal matrix $\tilde{\mathbf{D}}$ and an approximately orthogonal matrix $\tilde{\mathbf{U}}$ such that

$$\left\| \tilde{\mathbf{U}}^\top \tilde{\mathbf{U}} - \mathbf{I} \right\| \leq 3(\epsilon_1 + \epsilon)/n, \quad \text{and} \quad \left\| \mathbf{T} - \tilde{\mathbf{U}} \tilde{\mathbf{D}} \tilde{\mathbf{U}}^\top \right\| \leq 2\epsilon_1 + 7\epsilon,$$

in a total of $O\left(n^2 \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$ arithmetic operations and comparisons, where $\xi \geq 1$ is a small constant that depends on the specific FMM implementation and it is independent of ϵ, n .

Proof. The proof is deferred in the Appendix (see the restatement in Lemma C.1). \square

Lemma 2.1 gives rise to the following recursive algorithm. We can finally proceed with the proof of Theorem 1.1, which gives the complexity of Algorithm 1.

Algorithm: $[\tilde{\mathbf{U}}, \tilde{\mathbf{A}}] \leftarrow \text{DIAGONALIZE}(\mathbf{T}, \epsilon)$

- 1: **if** $n \leq 2$:
- 2: Compute $\tilde{\mathbf{U}}, \tilde{\mathbf{A}}$ to be the exact diagonalization of \mathbf{T} .
- 3: **else:**
- 4: Partition $\mathbf{T} = \begin{pmatrix} \mathbf{T}_1 & \beta_{k+1}\mathbf{e}_k & \\ \beta_{k+1}\mathbf{e}_k^\top & \alpha_{k+1} & \beta_{k+2}\mathbf{e}_1^\top \\ & \beta_{k+2}\mathbf{e}_1 & \mathbf{T}_2 \end{pmatrix}$.
- 5: $[\tilde{\mathbf{U}}_1, \tilde{\mathbf{D}}_1] \leftarrow \text{DIAGONALIZE}(\mathbf{T}_1, \epsilon)$.
- 6: $[\tilde{\mathbf{U}}_2, \tilde{\mathbf{D}}_2] \leftarrow \text{DIAGONALIZE}(\mathbf{T}_2, \epsilon)$.
- 7: Assemble $\tilde{\mathbf{U}}, \tilde{\mathbf{A}}$ from $\mathbf{T}, \tilde{\mathbf{U}}_1, \tilde{\mathbf{D}}_1, \tilde{\mathbf{U}}_2, \tilde{\mathbf{D}}_2$ using Lemma 2.1 with parameter ϵ .
- 8: **return** $\tilde{\mathbf{U}}, \tilde{\mathbf{A}}$.

Algorithm 1: Recursive algorithm based on [48] to diagonalize a symmetric tridiagonal matrix.

Proof of Theorem 1.1. Let $\mathbf{T}^{(p)}$ be a matrix at recursion depth p . We can always divide $\mathbf{T}^{(p)}$ such that the sizes of $\mathbf{T}_{\{1,2\}}^{(p)}$ differ by at most 1. If we keep dividing the matrix $\mathbf{T}^{(p)}$ in this manner, then the recursion tree has depth at most $d = \lceil \log(n) \rceil$.

The correctness follows by induction. Consider the base case d where $\mathbf{T}^{(d)}$ has size at most 5×5 , which means that $\mathbf{T}_{1,2}^{(d)}$ have size 1×1 or 2×2 . We can compute the exact diagonalization of $\mathbf{T}_{1,2}^{(d)}$, and therefore they satisfy the requirements of Lemma 2.1 for $\epsilon_1 = 0$. Thus, if we apply Lemma 2.1 with parameter ϵ' to compute the matrices $\tilde{\mathbf{U}}^{(d)}$ and $\tilde{\mathbf{D}}^{(d)}$, they will satisfy the following bounds:

$$\begin{aligned} \text{err}_{\tilde{\mathbf{U}}}(d) &:= \left\| \tilde{\mathbf{U}}^{(d)\top} \tilde{\mathbf{U}}^{(d)} - \mathbf{I} \right\| \leq 3(\epsilon_1 + \epsilon')/n = 3\epsilon'/n, \\ \text{err}_{\mathbf{T}}(d) &:= \left\| \mathbf{T} - \tilde{\mathbf{U}}^{(d)} \tilde{\mathbf{D}}^{(d)} \tilde{\mathbf{U}}^{(d)\top} \right\| \leq 7\epsilon'. \end{aligned}$$

We need to bound the error at the root (depth $p = 0$). As long as the error at depth $p - 1$ is smaller than $1/2$, the error at depth p is bounded by

$$\begin{aligned} \text{err}_{\tilde{\mathbf{U}}}(p) &\leq 3\text{err}_{\tilde{\mathbf{U}}}(p+1) + 3\epsilon'/n < 4\text{err}_{\tilde{\mathbf{U}}}(p+1) + 4\epsilon'/n, \\ \text{err}_{\mathbf{T}}(p) &\leq 2\text{err}_{\mathbf{T}}(p+1) + 7\epsilon' < 8\text{err}_{\mathbf{T}}(p+1) + 8\epsilon'. \end{aligned}$$

Solving the recursion we have

$$\text{err}_{\tilde{\mathbf{U}}}(0) < \sum_{p=1}^{\lceil \log(n) \rceil} 4^p \epsilon'/n = O(n^2) \frac{\epsilon'}{n} = O(n) \epsilon', \quad \text{err}_{\mathbf{T}}(0) < \sum_{p=1}^{\lceil \log(n) \rceil} 8^p \epsilon' = O(n^3) \epsilon'.$$

It thus suffices to run the algorithm with initial error $\epsilon' = c\epsilon/n^3$ for some small constant c .

The complexity analysis for ϵ' follows. For size n , the complexity is given by

$$T(n) \leq 2T(\frac{n}{2}) + Cn^2 \log^{\xi+1}(\frac{n}{\epsilon'}).$$

Solving the recursion yields $T(n) = O\left(n^2 \log^{\xi+1}(\frac{n}{\epsilon'})\right) = O\left(n^2 \log^{\xi+1}(\frac{n}{\epsilon})\right)$.

The final matrices $\tilde{\mathbf{U}} = \tilde{\mathbf{U}}^{(0)}$ and $\tilde{\mathbf{\Lambda}} = \tilde{\mathbf{D}}^{(0)}$ satisfy $\|\tilde{\mathbf{U}}^{(d)\top} \tilde{\mathbf{U}}^{(d)} - \mathbf{I}\| \leq \epsilon/n^2$ and $\|\mathbf{T} - \tilde{\mathbf{U}} \tilde{\mathbf{\Lambda}} \tilde{\mathbf{U}}^\top\| \leq \epsilon$, which means that:

$$\|\tilde{\mathbf{\Lambda}}\| \leq \|\tilde{\mathbf{U}}^{-1}\|^2 \left(\|\mathbf{T}\| + \|\mathbf{T} - \tilde{\mathbf{U}} \tilde{\mathbf{\Lambda}} \tilde{\mathbf{U}}^\top\| \right) \leq \frac{1}{1 - \epsilon/n^2} (1 + \epsilon).$$

We can then write $\tilde{\mathbf{U}} = \mathbf{U} + \mathbf{E}_U$, which gives

$$\begin{aligned} \|\mathbf{T} - \mathbf{U} \tilde{\mathbf{\Lambda}} \mathbf{U}^\top\| &\leq \|\mathbf{T} - \tilde{\mathbf{U}} \tilde{\mathbf{\Lambda}} \tilde{\mathbf{U}}^\top\| + 2\|\tilde{\mathbf{\Lambda}}\| \|\mathbf{E}_U\| + \|\tilde{\mathbf{\Lambda}}\| \|\mathbf{E}_U\|^2 \\ &\leq \epsilon + \frac{(1+\epsilon)}{1-\epsilon/n^2} \frac{\epsilon}{n^2} + \frac{1+\epsilon}{1-\epsilon/n^2} \left(\frac{\epsilon}{n^2}\right)^2 = \epsilon \left(1 + \frac{(1+\epsilon)(1+\epsilon/n^2)}{n^2 - \epsilon}\right). \end{aligned}$$

Rescaling ϵ by a small constant slightly larger than one gives the alternative statement. \square

2.3 Hermitian diagonalization

Given an algorithm to diagonalize tridiagonal matrices, the following corollary is immediate.

Corollary 2.1. *Let \mathbf{A} be a Hermitian matrix of size n with $\|\mathbf{A}\| \leq 1$. Given accuracy $\epsilon \in (0, 1/2)$, and an (ϵ, n) -FMM implementation of Prop. 1.1, we can compute a matrix $\tilde{\mathbf{Q}}$ and a diagonal matrix $\tilde{\mathbf{\Lambda}}$ such that*

$$\|\mathbf{A} - \tilde{\mathbf{Q}} \tilde{\mathbf{\Lambda}} \tilde{\mathbf{Q}}^*\| \leq \epsilon, \quad \|\tilde{\mathbf{Q}}^* \tilde{\mathbf{Q}} - \mathbf{I}\| \leq \epsilon/n^2.$$

The algorithm requires a total of $O\left(n^\omega \log(n) + n^2 \log^{\xi+1}(\frac{n}{\epsilon})\right)$ arithmetic operations and comparisons, where $\xi \geq 1$ is a small constant that depends on the specific FMM implementation and it is independent of ϵ, n .

Proof. A full proof, as well as an alternative statement of the result, can be found in the restatement, Corollary C.2, in Appendix C.3. \square

3 Stability of tridiagonal reduction

In this section we analyze the numerical stability and the boolean complexity of Schönhage's algorithm the floating point model. For this we will use the following stable matrix multiplication and backward-stable QR factorization algorithms as subroutines from [28, 29]. The corresponding definitions and imported results for these subroutines are deferred to Appendix E.1.

3.1 Matrix nomenclature

Schönhage [79] used a block variant of Rutishauser's algorithm [76] to reduce a matrix to tridiagonal form, where elementary rotations are replaced with block factorizations; see also [16, 4, 5] for similar methodologies. We start with a $n \times n$ block-pentadiagonal matrix $\mathbf{A}^{(k,s,t)}$, where $k \in \{0, \dots, \log(n) - 2\}$ (we assume without loss of generality that n is a power of two). The matrix $\mathbf{A}^{(k,s,t)}$ is partitioned in $b_k \times b_k$ blocks of size $n_k \times n_k$ each, where $b_k = \frac{n}{n_k}$ and $n_k = 2^k$. The integer $s \in 2, \dots, b_k$ denotes that all the blocks $\mathbf{A}_{i,i-2}$ and $\mathbf{A}_{i-2,i}$, for all $i = 2, \dots, s$ are equal to zero. $s = 2$ is a special case to denote a full block pentadiagonal matrix. The integer $t \in \{s+2, \dots, b_k\}$ denotes that the matrix has two additional nonzero blocks in the third off-diagonals, specifically at positions $\mathbf{A}_{t,t-3}$ and $\mathbf{A}_{t-3,t}$. These blocks are often called the "bulge" in the literature. When $t = 0$, there is no bulge. As a consequence, the matrix

$A^{(k,2,0)}$ is full block-pentadiagonal, while the matrix $A^{(k,b_k,0)}$ is block-tridiagonal. An illustration of these definitions is shown in Equation (4). A box is placed around the bulge on the second matrix.

$$\begin{array}{c}
 \overbrace{A^{(k,2,0)}} \\
 \left(\begin{array}{cccccccc}
 A_{1,1} & A_{1,2} & A_{1,3} & 0 & 0 & 0 & 0 & 0 \\
 A_{2,1} & A_{2,2} & A_{2,3} & A_{2,4} & 0 & 0 & 0 & 0 \\
 A_{3,1} & A_{3,2} & A_{3,3} & A_{3,4} & A_{3,5} & 0 & 0 & 0 \\
 0 & A_{4,2} & A_{4,3} & A_{4,4} & A_{4,5} & A_{4,6} & 0 & 0 \\
 0 & 0 & A_{5,3} & A_{5,4} & A_{5,5} & A_{5,6} & A_{5,7} & 0 \\
 0 & 0 & 0 & A_{6,4} & A_{6,5} & A_{6,6} & A_{6,7} & A_{6,8} \\
 0 & 0 & 0 & 0 & A_{7,5} & A_{7,6} & A_{7,7} & A_{7,8} \\
 0 & 0 & 0 & 0 & 0 & A_{8,6} & A_{8,7} & A_{8,8}
 \end{array} \right),
 \end{array}
 \begin{array}{c}
 \overbrace{A^{(k,4,6)}} \\
 \left(\begin{array}{cccccccc}
 A_{1,1} & A_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 \\
 A_{2,1} & A_{2,2} & A_{2,3} & 0 & 0 & 0 & 0 & 0 \\
 0 & A_{3,2} & A_{3,3} & A_{3,4} & A_{3,5} & \boxed{A_{3,6}} & 0 & 0 \\
 0 & 0 & A_{4,3} & A_{4,4} & A_{4,5} & A_{4,6} & 0 & 0 \\
 0 & 0 & A_{5,3} & A_{5,4} & A_{5,5} & A_{5,6} & A_{5,7} & 0 \\
 0 & 0 & \boxed{A_{6,3}} & A_{6,4} & A_{6,5} & A_{6,6} & A_{6,7} & A_{6,8} \\
 0 & 0 & 0 & 0 & A_{7,5} & A_{7,6} & A_{7,7} & A_{7,8} \\
 0 & 0 & 0 & 0 & 0 & A_{8,6} & A_{8,7} & A_{8,8}
 \end{array} \right).
 \end{array}
 \quad (4)$$

3.2 Rotations

The algorithm defines two types of block rotations R_i and R'_j , which are unitary similarity transformations, with the following properties.

Definition 3.1 (Rotations). *The algorithm of [79] uses the following two types of rotations:*

1. $R_i(A^{(k,i,0)})$, for $i = 2, \dots, b_k - 1$, operates on a block-pentadiagonal matrix without a bulge. It transforms the matrix $A^{(k,i,0)}$ to a matrix $A^{(k,i+1,i+3)}$. In particular, the block $A_{i,i-1}$ becomes upper triangular, the block $A_{i+1,i-1}$ becomes zero, and a new bulge block arises at $A_{i,i+3}$. Due to symmetry, $A_{i-1,i}$ becomes lower triangular, $A_{i-1,i+1}$ is eliminated, and $A_{i+3,i}$ becomes non-zero.
2. $R'_j(A^{(k,s,j+1)})$, for some $j = s + 1, \dots, b_k - 1$, operates on a block-pentadiagonal matrix with a bulge at positions $A_{j+1,j-2}$, $A_{j-2,j+1}$. It transforms the matrix $A^{(k,s,j+1)}$ to a matrix $A^{(k,s,j+3)}$, such that the bulge is moved two positions “down-and-right”, i.e. the blocks $A_{j-2,j+1}$ and $A_{j+1,j-2}$ become zero and the blocks $A_{j,j+3}$ and $A_{j+3,j}$ become the new bulge. In addition, the matrices $A_{j,j-2}$ and $A_{j+1,j-1}$ become upper triangular, and, by symmetry, the matrices $A_{j-2,j}$ and $A_{j-1,j+1}$ become lower triangular.

An example of the aforementioned rotations is illustrated in Equations (36) and (37) in the Appendix, and in Lemmas E.1 and E.2 it is proved that both types can be stably implemented in floating point using fast QR factorizations.

3.3 Recursive bandwidth halving

Using Lemmas E.1 and E.2, we can analyze the following Algorithm 2, which halves the bandwidth of a matrix. Its complexity and stability properties are stated in Lemma E.3 in the Appendix. Applying this algorithm recursively gives the main Theorem 3.1.

Algorithm: $[\widehat{Q}^{(k)}, A^{(k-1,2,0)}] \leftarrow \text{HALVE}(A^{(k,2,0)}, k, n)$

- 1: Set $n_k = 2^k$, $b_k = n/n_k$.
- 2: **for** $i = 2, \dots, b_k$:
- 3: Compute $Q_{i,i}$, $A^{(k,i+1,i+3)} \leftarrow R_i(A^{(k,i,0)})$.
- 4: **for** $j = i + 2, \dots, b_k$ with step 2 :
- 5: Compute $Q_{i,j}$, $A^{(k,i+1,j+3)} \leftarrow R'_j(A^{(k,i+1,j+1)})$.
- 6: Stack together all the matrices $Q_{i,j}$ to form Q_i .
- 7: Assemble the matrix $\widehat{Q}^{(k)}$ by multiplying the matrices Q_i .
- 8: **return** $\widehat{Q}^{(k)}, A^{(k-1,2,0)}$.

Algorithm 2: Halves the bandwidth of a Hermitian matrix with unitary rotations.

Theorem 3.1. *There exists a floating point implementation of the tridiagonal reduction algorithm of [79], which takes as input a Hermitian matrix \mathbf{A} , and returns a tridiagonal matrix $\tilde{\mathbf{T}}$, and (optionally) an approximately unitary matrix $\tilde{\mathbf{Q}}$. If the machine precision \mathbf{u} satisfies $\mathbf{u} \leq \epsilon \frac{1}{cn^{\beta+4}}$, where $\epsilon \in (0, 1)$, c is a constant, and β is the same as in Corollary E.1, which translates to $O(\log(n) + \log(1/\epsilon))$ bits of precision, then the following hold:*

$$\|\tilde{\mathbf{Q}}\tilde{\mathbf{Q}}^* - \mathbf{I}\| \leq \epsilon, \quad \text{and} \quad \|\mathbf{A} - \tilde{\mathbf{Q}}\tilde{\mathbf{T}}\tilde{\mathbf{Q}}^*\| \leq \epsilon \|\mathbf{A}\|.$$

The algorithm executes at most $O(n^2 S_\omega(\log(n)))$ floating point operations to return only $\tilde{\mathbf{T}}$, where $S_x(m) = \sum_{l=1}^m (2^{x-2})^l$. If \mathbf{A} is banded with $1 \leq d \leq n$ bands, the floating point operations reduce to $O(n^2 S_\omega(\log(d)))$. If $\tilde{\mathbf{Q}}$ is also returned, the complexity increases to $O(n^2 C_\omega(\log(n)))$, where $C_x(n) := \sum_{k=2}^{\log(n)-2} (S_x(\log(n) - 1) - S_x(k))$. If ω is treated as a constant $\omega \approx 2.371$ the corresponding complexities are $O(n^\omega)$, $O(n^2 d^{\omega-2})$, and $O(n^\omega \log(n))$, respectively.

Proof. The proof is deferred to the Appendix (see the restatement in Theorem E.3). \square

3.4 Eigenvalues of Hermitian matrices

We now have all the prerequisites to compute the eigenvalues Hermitian matrices in nearly matrix multiplication time in finite precision. For this we can use the eigenvalue solver of [13], which has $\tilde{O}(n^2)$ boolean complexity, albeit in the Boolean RAM model. Specifically, the algorithm accepts as input symmetric tridiagonal matrices with bounded integer entries.

Theorem 3.2 (Imported from [13, 15]). *Let \mathbf{T} be a symmetric tridiagonal matrix with integer elements bounded in magnitude by 2^m for some m . Let $\epsilon = 2^{-u} \in (0, 1)$ be a desired accuracy. Algorithm 4.1 of [13] computes a set of approximate eigenvalues $\tilde{\lambda}_i \in \mathbb{R}$ (which are returned as rationals) such that $|\tilde{\lambda}_i - \lambda_i(\mathbf{T})| < \epsilon$. The algorithm requires $O(n^2 b \log^2(n) \log(nb) (\log^2(b) + \log(n)) \log(\log(nb)))$ boolean operations, where $b = m + u$.*

Theorem 3.3. *Let \mathbf{A} be a (banded) Hermitian matrix, with $\|\mathbf{A}\| \leq 1$, $1 \leq d \leq n - 1$ off-diagonals, and let $\epsilon \in (0, 1)$ be an accuracy parameter. Assume that the elements of \mathbf{A} are floating point numbers on a machine with precision \mathbf{u} , $t = \log(1/\mathbf{u})$ bits for the significand, and $p = O(\log(\log(n)))$ bits for the exponent. There exists an algorithm that returns a set of n approximate eigenvalues $\lambda_1, \dots, \lambda_n$ such that*

$$|\tilde{\lambda}_i - \lambda_i(\mathbf{A})| \leq \epsilon$$

using at most

$$O(n^2 S_\omega(\log(d)) \cdot \mathcal{F}(\log(\frac{n}{\epsilon})) + n^2 \text{polylog}(\frac{n}{\epsilon}))$$

boolean operations, where $\mathcal{F}(b)$ is the bit complexity of a floating point operation on b bits, and $n^2 S_\omega(\log(d)) = O(n^2 d^{\omega-2})$ if ω is treated as a constant greater than two.

Proof. First, recall that each element of a floating point matrix \mathbf{A} is represented by a floating point number $\mathbf{A}_{i,j} = \pm 1 \times 2^{e_{i,j}} \times m_{i,j}$, where $e_{i,j} \in [-2^p, 2^p]$ is its exponent, p is the number of exponent bits, and $m_{i,j}$ is the significand, which is an integer with $t = \log(1/\mathbf{u})$ bits. To represent all the elements of \mathbf{A} with normalized floating point numbers it is sufficient to use $p = O(\log(\log(A)))$, where $A = \max \left\{ \|\mathbf{A}\|_{\max}, \frac{1}{\|\mathbf{A}\|_{\min}} \right\}$. By assumption, $\|\mathbf{A}\|_{\max} \leq \|\mathbf{A}\| \leq 1$. It is common to assume that $\frac{1}{\|\mathbf{A}\|_{\min}} \in \text{poly}(n)$, which means that $O(\log(\log(n)))$ bits for the exponent are sufficient.

To compute the eigenvalues of \mathbf{A} we work as follows. We first run the algorithm of Theorem 3.1 to reduce \mathbf{A} to tridiagonal form $\tilde{\mathbf{T}}$, with parameter ϵ and $O(\log(\frac{n}{\epsilon}))$ bits of precision, and a total of $O(n^2 S_\omega(\log(d)))$ floating point operations, which is equal to $O(n^2 d^{\omega-2})$ if ω is treated as a constant. It holds that

$$|\lambda_i(\tilde{\mathbf{T}}) - \lambda_i(\mathbf{A})| \leq \epsilon' \|\mathbf{A}\| \leq \epsilon \frac{\|\mathbf{A}\|}{n 2^{2^{p+1}}} \leq \epsilon,$$

where in the last inequality we used that $\|\mathbf{A}\| \leq n \|\mathbf{A}_{\max}\| \leq n 2^{2^{p+1}}$.

Therefore, it suffices to approximate the eigenvalues of $\tilde{\mathbf{T}}$. For this we first transform $\tilde{\mathbf{T}}$ to a tridiagonal matrix with integer values and use the algorithm of [13] to compute its eigenvalues. A floating point matrix can be transformed to one with integer entries as follows. Assuming that we have p bits for the floating point exponent, and $t = \log(1/\mathbf{u})$ bits for the significand, the floating point numbers with $O(p + t)$ bits can be transformed to integers of $O(2^{2p} + t)$ bits by multiplying all the elements with 2^{2p+t} . This is achieved by first allocating a tridiagonal matrix with $3n - 2$ integers with $O(2^{2p} + t) = O(\log^2(n) + \log(n/\epsilon))$ bits each, that are initially set to zero. We then visit each floating point element $\tilde{\mathbf{T}}_{i,j}$ of the original matrix, which is represented as a number $\pm 1 \times 2^{e_{i,j}} \times m_{i,j}$, where $e_{i,j} \in [-2^p, 2^p]$ is its exponent and $m_{i,j}$ is the significand, which is an integer with t bits. We copy the t bits of $m_{i,j}$ at the positions $e_{i,j} + 2^p, e_{i,j} + 2^p + 1, \dots, e_{i,j} + 2^p + t$ of the corresponding element of the new matrix. This takes $O(n(2^{2p} + t)) = O(n(\log^2(n) + \log(n/\epsilon)))$ boolean operations in total to allocate the new matrix and copy the elements.

Now that we have an integer valued symmetric tridiagonal matrix \mathbf{T}' , with $O(2^{2p} + t) = O(\log^2(n) + \log(n/\epsilon))$ bits per element, we can compute its eigenvalues up to additive accuracy ϵ with Theorem 3.2. Specifically, we set $\epsilon'' = \epsilon \cdot 2^{2p+t}$, and run the algorithm with the matrix \mathbf{T}' as input and $u = \log(\frac{1}{\epsilon''})$. Let

$$b = u + m = \log\left(\frac{1}{\epsilon''}\right) + 2^{p+1} + t = \log\left(\frac{1}{\epsilon 2^{2p+t}}\right) + \log\left(2^{2^{p+1}+t}\right) = O(\log(\frac{n}{\epsilon})),$$

where the last inequality follows from the assumption that $p = O(\log(\log(n)))$. The returned eigenvalues λ'_i satisfy

$$|\lambda'_i - \lambda_i(\mathbf{T}')| \leq \epsilon'' \Rightarrow \left| \frac{\lambda'_i}{2^{2p+t}} - \lambda_i(\tilde{\mathbf{T}}) \right| \leq \epsilon.$$

Therefore, if we set $\tilde{\lambda}_i = \frac{\lambda'_i}{2^{2p+t}}$, we finally obtain that $|\tilde{\lambda}_i - \lambda_i(\mathbf{A})| \leq 2\epsilon$. Rescaling ϵ gives the final bound. The algorithm runs in

$$\begin{aligned} B &= O\left(n^2 b \log^2(n) \log(nb) (\log^2(b) + \log(n)) \log(\log(nb))\right) \\ &= O\left(n^2 \log\left(\frac{n}{\epsilon}\right) \log^2(n) \log(n \log\left(\frac{n}{\epsilon}\right)) \left[\log^2(\log\left(\frac{n}{\epsilon}\right)) + \log(n)\right] \log(\log(n \log\left(\frac{n}{\epsilon}\right)))\right) \end{aligned}$$

boolean operations. □

4 Further applications of stable tridiagonal reduction and Hermitian eigenvalue solver

In this section we state some applications of the tridiagonal reduction algorithm to some eigenproblems.

4.1 Singular values and condition number

Proposition 4.1. *Given a matrix $\mathbf{A} \in \mathbb{C}^{n \times n}$, with $\|\mathbf{A}\| \leq 1$, an integer $k \in [n]$, and accuracy $\epsilon \in (0, 1)$, we can compute a value $\tilde{\sigma}_k \in (1 \pm \epsilon)\sigma_k(\mathbf{A})$ in*

$$O\left(\left[n^\omega \mathcal{F}(\log(\frac{n}{\epsilon\sigma_k})) + n^2 \text{polylog}(\frac{n}{\epsilon\sigma_k})\right] \log(\log(\epsilon\sigma_k))\right)$$

boolean operations, deterministically.

Note: If $\|\mathbf{A}\| > 1$, we can scale with $1/(n\|\mathbf{A}\|_{\max})$ or $1/\|\mathbf{A}\|_F$. The complexity is unaffected.

Proof. The problem is solved by iteratively calling Theorem 3.3 on $\mathbf{A}\mathbf{A}^*$, squaring the accuracy in each iteration until the desired singular value is approximated. First, we divide \mathbf{A} by two, to ensure that $\|\mathbf{A}\| \leq 1/2$. Then we construct $\tilde{\mathbf{A}} = \mathbf{M}\mathbf{M}(\mathbf{A}, \mathbf{A}^*) = \mathbf{A}\mathbf{A}^*/4 + \mathbf{E}^{\text{MM}}$. We start with $\epsilon_0 = 1/2$. From Weyl's inequality, $|\lambda_k(\tilde{\mathbf{A}}) - \lambda_k(\mathbf{A}\mathbf{A}^*/4)| \leq \epsilon_0/2$.

Next, we call Theorem 3.3 to compute all the eigenvalues of $\tilde{\mathbf{A}}$ up to accuracy $\epsilon_0/2$. The algorithm returns approximate eigenvalues $\tilde{\lambda}_1^{(0)}, \dots, \tilde{\lambda}_n^{(0)}$ such that

$$\left| \tilde{\lambda}_i^{(0)} - \lambda_i(\tilde{\mathbf{A}}) \right| \leq \epsilon_0/2 \quad \Rightarrow \quad \left| \tilde{\lambda}_i^{(0)} - \lambda_i(\mathbf{A}\mathbf{A}^*/4) \right| \leq \epsilon_0,$$

using at most $O\left(n^\omega \mathcal{F}(\log(\frac{n}{\epsilon_0})) + n^2 \text{polylog}(\frac{n}{\epsilon_0})\right)$ boolean operations.

We keep calling the above steps iteratively, by squaring ϵ_t in every iteration t , i.e., $\epsilon_t = \epsilon_0^{2^t}$. Let $\lambda_i = \lambda_i(\mathbf{A}\mathbf{A}^*/4)$. We run the algorithm until $\epsilon_t \leq \frac{\epsilon}{2} |\tilde{\lambda}_k^{(t)}|$. The latter condition ensures that

$$\epsilon_t \leq \frac{\epsilon}{2} |\tilde{\lambda}_k^{(t)}| \leq \frac{\epsilon}{2} (\lambda_k + \epsilon_t) \Rightarrow \epsilon_t \leq \epsilon \lambda_k,$$

where in the last we used the assumption that $\epsilon \in (0, 1/2)$. Since at each iteration $\lambda_k - \epsilon_t \leq \tilde{\lambda}_k^{(t)}$, then the following condition is sufficient for termination:

$$\epsilon_t \leq \frac{\epsilon}{2} (\lambda_k - \epsilon_t) \Rightarrow \epsilon_t \leq \frac{2}{5} \epsilon \lambda_k.$$

This is reached in at most $t = O(\log(\log(\epsilon \lambda_k)))$ iterations. From the cost of the matrix multiplications and from Theorem 3.3, each iteration costs

$$O\left(mn^{\omega-1} \mathcal{F}(\log(\frac{mn}{\epsilon_t})) + n^2 S_\omega(\log(n)) \cdot \mathcal{F}(\log(\frac{n}{\epsilon_t})) + n^2 \text{polylog}(n/\epsilon_t)\right),$$

boolean operations, which is maximized in the last iteration. This gives a total cost of at most

$$O\left(t \left[mn^{\omega-1} \mathcal{F}(\log(\frac{mn}{\epsilon \sigma_k})) + n^2 S_\omega(\log(n)) \mathcal{F}(\log(\frac{n}{\epsilon \sigma_k})) + \text{polylog}(\frac{n}{\epsilon \sigma_k}) \right]\right)$$

boolean operations, since $\lambda_k = \sigma_k^2/4$, which is equal to

$$O\left(\left[mn^{\omega-1} \mathcal{F}(\log(\frac{mn}{\epsilon \sigma_k})) + n^2 \text{polylog}(\frac{n}{\epsilon \sigma_k}) \right] \log(\log(\epsilon \sigma_k))\right)$$

for constant $\omega \approx 2.371$. We finally return $\tilde{\sigma}_k = 2\sqrt{\tilde{\lambda}_k^{(t)}}$, which satisfies the advertised bound. \square

Corollary 4.1. *Let $\mathbf{A} \in \mathbb{C}^{n \times n}$, $\kappa = \kappa(\mathbf{A})$, and $\delta \in (0, 1/2)$. We can compute $\tilde{\kappa}$ which $\kappa \leq \tilde{\kappa} \leq 3n\kappa$, in*

$$O\left(\left[n^\omega \mathcal{F}(\log(n\kappa)) + n^2 \text{polylog}(n\kappa)\right] \log(\log(n\kappa))\right)$$

boolean operations deterministically.

Proof. We first set $\tilde{\Sigma} = n\|\mathbf{A}\|_{\max}$. Assuming $O(\log(n))$ bits for the elements of \mathbf{A} , this takes $O(n^2 \mathcal{F}(\log(n)))$ boolean operations (for floating point comparisons). We can then scale $\mathbf{A}' \leftarrow \mathbf{A}/M$, where M is again the smallest power of 2 that is larger than $2\tilde{\Sigma}$. This gives $\frac{1}{2n} \leq \|\mathbf{A}'\| \leq \frac{1}{2}$, and also $\sigma_{\min}(\mathbf{A}') \in \left[\frac{\sigma_{\min}(\mathbf{A})}{2n\|\mathbf{A}\|}, \frac{\sigma_{\min}(\mathbf{A})}{2\|\mathbf{A}\|}\right] = \left[\frac{1}{2n\kappa(\mathbf{A})}, \frac{1}{2\kappa(\mathbf{A})}\right]$. As in the previous case, we call Proposition 4.1 on \mathbf{A}' with $k = 1$ and error $\epsilon = 1/2$, which returns $\tilde{\sigma}'_{\min} \in (1 \pm \frac{1}{2})\sigma_{\min}(\mathbf{A}')$. It requires

$$O\left(\left[n^\omega \mathcal{F}(\log(n\kappa(\mathbf{A}))) + n^2 \text{polylog}(n\kappa(\mathbf{A})))\right] \log(\log(n\kappa(\mathbf{A})))\right)$$

boolean operations, and we finally set $\tilde{\kappa} = \frac{1}{\tilde{\sigma}'_{\min}} \in [\kappa(\mathbf{A}), 3n\kappa(\mathbf{A})]$. \square

4.2 Definite pencil eigenvalues

Using the proposed algorithms we can compute the eigenvalues of a Hermitian definite pencil.

Corollary 4.2. *Let \mathbf{H} be Hermitian, \mathbf{S} Hermitian positive-definite, both with size n and floating point elements, on a machine with precision \mathbf{u} , $t = \log(1/\mathbf{u})$ bits for the significand, and $p = O(\log(\log(n)))$ bits for the exponent. Assume that $\|\mathbf{H}\|, \|\mathbf{S}^{-1}\| \leq 1$, $\kappa(\mathbf{S}) \in \text{poly}(n)$, and that we have access to $\tilde{\kappa} \in [\kappa(\mathbf{S}), Z\kappa(\mathbf{S})]$, where $Z > 1$ might be a constant or a function of n . There exists an algorithm that returns a set of n approximate eigenvalues $\tilde{\lambda}_1, \dots, \tilde{\lambda}_n$ such that*

$$|\tilde{\lambda}_i - \lambda_i(\mathbf{H}, \mathbf{S})| \leq \epsilon$$

using at most

$$O\left(n^\omega \mathcal{F}(\log(n) \log(Z\kappa(\mathbf{S})) + \log(\frac{1}{\epsilon})) + n^2 \text{polylog}(\frac{n}{\epsilon})\right)$$

boolean operations. If $\tilde{\kappa}$ is not given, it can be computed up to a factor $Z = 3n$ with Corollary 4.1.

Proof. We can use [83, Prop. C.3] with error $\epsilon/2$, which requires $O(n^\omega)$ floating point operations and a total of $O(\log(n) \log(Z\kappa(\mathbf{S})) + \log(\frac{1}{\epsilon}))$ bits, and it returns a Hermitian matrix $\tilde{\mathbf{H}}$ that satisfies $|\lambda_i(\tilde{\mathbf{H}}) - \lambda_i(\mathbf{H}, \mathbf{S})| \leq \frac{\epsilon}{2}$.

Then, Theorem 3.3 is used with parameter $\epsilon/2$ to compute the eigenvalues of $\tilde{\mathbf{H}}$ up to additive error $\epsilon/2$. We can use it if $\kappa(\mathbf{S}) \in \text{poly}(n)$, otherwise the problem is ill-conditioned and we might need more than $O(\log(\log(n)))$ bits for the floating point exponent. Regarding the bits of precision $O(\log(n/\epsilon))$ bits are sufficient for Theorem 3.3. But this is already covered, since we assumed that we have at least $O(\log(n) \log(Z\kappa(\mathbf{S})) + \log(1/\epsilon))$ bits of precision in the previous step. Then the complexity of Theorem 3.3 for the increased number of bits becomes

$$O\left(n^\omega \mathcal{F}(\log(n) \log(Z\kappa(\mathbf{S})) + \log(\frac{1}{\epsilon})) + n^2 \text{polylog}(\frac{n}{\epsilon})\right)$$

boolean operations. The returned eigenvalues $\tilde{\lambda}_1, \dots, \tilde{\lambda}_n$ satisfy

$$|\tilde{\lambda}_i - \lambda_i(\mathbf{H}, \mathbf{S})| \leq |\tilde{\lambda}_i - \lambda_i(\tilde{\mathbf{H}})| + |\lambda_i(\tilde{\mathbf{H}}) - \lambda_i(\mathbf{H}, \mathbf{S})| \leq 2\epsilon/2 = \epsilon.$$

□

Remark 4.1. *In Theorem 4.2 we assumed that $\|\mathbf{H}\|, \|\mathbf{S}^{-1}\| \leq 1$. This is not a limitation since we can approximate $\|\mathbf{S}^{-1}\|$ with Proposition 4.1, and then scale accordingly. Formally, let $\eta \gtrsim \|\mathbf{H}\|$ and $\sigma \gtrsim \|\mathbf{S}^{-1}\|$. Then we can rewrite the generalized eigenproblem*

$$\mathbf{H}\mathbf{C} = \mathbf{S}\mathbf{C}\mathbf{A} \Leftrightarrow (\frac{1}{\eta}\mathbf{H})\mathbf{C} = (\sigma\mathbf{S})\mathbf{C}(\frac{1}{\eta\sigma}\mathbf{A}) \Leftrightarrow \mathbf{H}'\mathbf{C} = \mathbf{S}'\mathbf{C}\mathbf{A}',$$

i.e. it is the same generalized eigenproblem only with scaled eigenvalues. Assuming that the matrices \mathbf{H} and \mathbf{S} are “well-conditioned,” i.e. their norms and condition numbers $\in \text{poly}(n)$, the eigenvalues are scaled by at most a $1/\text{poly}(n)$ factor, and thus it suffices to scale ϵ by $1/\text{poly}(n)$ as well. We can thus safely make the unit-norms assumption.

4.3 Spectral gaps

In a similar way we can compute the spectral gap between a pair of eigenvalues of Hermitian matrices and Hermitian-definite pencils.

Corollary 4.3. *Let \mathbf{A} be a banded Hermitian matrix of size n with $1 \leq d \leq n-1$ off-diagonals, $\|\mathbf{A}\| \leq 1$, and its eigenvalues $\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$. For some integer $k \in [n-1]$, let $\mu_k = \frac{\lambda_k + \lambda_{k+1}}{2}$ and $\Delta_k = \lambda_{k+1} - \lambda_k$. Given an accuracy $\epsilon \in (0, 1)$, we can compute two values $\tilde{\mu}_k$ and $\tilde{\Delta}_k$ such that*

$$\tilde{\mu}_k \in \mu_k \pm \epsilon \Delta_k, \quad \text{and} \quad \tilde{\Delta}_k \in (1 \pm \epsilon) \Delta_k,$$

in

$$O\left(n^2 \left[d^{\omega-2} \cdot \mathcal{F}(\log(\frac{n}{\epsilon \Delta_k})) + \text{polylog}(\frac{n}{\epsilon \Delta_k}) \right] \log(\log(\frac{1}{\epsilon \Delta_k})) \right)$$

boolean operations.

Proof. We start with an initial $\epsilon_0 = 1/2$, and we call the algorithm of Theorem 3.3 to compute the eigenvalues of \mathbf{A} up to additive error ϵ_0 . We keep calling the algorithm iteratively, by squaring ϵ_t in every iteration t , i.e., $\epsilon_t = \epsilon_0^{2^t}$. In each iteration, the algorithm returns eigenvalues $\tilde{\lambda}_i^{(t)}$ that satisfy

$$\left| \tilde{\lambda}_i^{(t)} - \lambda_i(\mathbf{A}) \right| \leq \epsilon_t.$$

We run the algorithm until $\epsilon_t \leq \frac{\epsilon}{2} \tilde{\Delta}_k^{(t)} = \frac{\epsilon}{2} \left| \tilde{\lambda}_{k+1}^{(t)} - \tilde{\lambda}_k^{(t)} \right|$. The latter condition ensures that

$$\epsilon_t \leq \frac{\epsilon}{2} \left| \tilde{\lambda}_{k+1}^{(t)} - \tilde{\lambda}_k^{(t)} \right| \leq \frac{\epsilon}{2} (|\lambda_{k+1} - \lambda_k| + 2\epsilon_t) \Rightarrow \epsilon_t \leq \epsilon \Delta_k,$$

where in the last we used the assumption that $\epsilon \in (0, 1/2)$. Since at each iteration $\Delta_k - 2\epsilon_t \leq \tilde{\Delta}_k^{(t)}$, then the following condition is sufficient for termination:

$$\epsilon_t \leq \frac{\epsilon}{2} \Delta_k - 2\epsilon_t \Rightarrow \epsilon_t \leq \epsilon \frac{\Delta_k}{6}.$$

This is reached in at most $t = O(\log(\log(\frac{1}{\epsilon \Delta_k})))$ iterations. From Theorem 3.3, each iteration costs

$$O\left(n^2 S_\omega(\log(d)) \cdot \mathcal{F}(\log(\frac{n}{\epsilon_t})) + n^2 \text{polylog}(\frac{n}{\epsilon_t})\right),$$

boolean operations, which is maximized in the last iteration. This gives a total cost of at most

$$O\left(n^2 \log(\log(\frac{1}{\epsilon \Delta_k})) \left[S_\omega(\log(d)) \cdot \mathcal{F}(\log(\frac{n}{\epsilon \Delta_k})) + \text{polylog}(\frac{n}{\epsilon \Delta_k}) \right] \right)$$

boolean operations, which is equal to

$$O\left(n^2 \log(\log(\frac{1}{\epsilon \Delta_k})) \left[d^{\omega-2} \log(n) \mathcal{F}(\log(\frac{n}{\epsilon \Delta_k})) + \text{polylog}(\frac{n}{\epsilon \Delta_k}) \right] \right)$$

for constant $\omega \approx 2.371$. We finally return $\tilde{\Delta}_k = \tilde{\lambda}_{k+1}^{(t)} - \tilde{\lambda}_k^{(t)}$ and $\tilde{\mu}_k = \frac{\tilde{\lambda}_{k+1}^{(t)} + \tilde{\lambda}_k^{(t)}}{2}$ which satisfy the advertised bounds. \square

Corollary 4.4. *Let \mathbf{H} be Hermitian, \mathbf{S} Hermitian positive-definite, both with size n and $\|\mathbf{H}\|, \|\mathbf{S}^{-1}\| \leq 1$, which define a Hermitian-definite pencil (\mathbf{H}, \mathbf{S}) . Given $k \in [n-1]$, $\tilde{\kappa} \in [\kappa(\mathbf{S}), Z\kappa(\mathbf{S})]$, where $Z > 1$ might be a constant or a function of n , and accuracy $\epsilon \in (0, 1/2)$, we can compute $\tilde{\mu}_k = \mu_k \pm \epsilon \Delta_k$ and $\tilde{\Delta}_k = (1 \pm \epsilon) \Delta_k$, where $\mu_k = \frac{\lambda_k + \lambda_{k+1}}{2}$ and $\Delta_k = \lambda_k - \lambda_{k+1}$. The algorithm requires*

$$O\left(\left[n^\omega \mathcal{F}\left(\log(n) \log(Z\kappa(\mathbf{S})) + \log(\frac{1}{\epsilon \Delta_k})\right) + n^2 \text{polylog}(\frac{n}{\epsilon \Delta_k}) \right] \log(\log(\frac{1}{\epsilon \Delta_k})) \right)$$

boolean operations. If $\tilde{\kappa}$ is not given, it can be computed up to a factor $Z = 3n$ with Corollary 4.1.

Proof. We start with an initial $\epsilon_0 = 1/2$, and we call the algorithm of Theorem 4.2 to compute the eigenvalues of (\mathbf{H}, \mathbf{S}) up to additive error ϵ_0 . We keep calling the algorithm iteratively, by squaring ϵ_t in every iteration t , i.e., $\epsilon_t = \epsilon_0^{2^t}$. In each iteration, the algorithm returns eigenvalues $\tilde{\lambda}_i^{(t)}$ that satisfy

$$\left| \tilde{\lambda}_i^{(t)} - \lambda_i(\mathbf{H}, \mathbf{S}) \right| \leq \epsilon_t.$$

We terminate when $\epsilon_t \leq \frac{\epsilon}{2} \tilde{\Delta}_k^{(t)} = \frac{\epsilon}{2} \left| \tilde{\lambda}_{k+1}^{(t)} - \tilde{\lambda}_k^{(t)} \right|$. This ensures that

$$\epsilon_t \leq \frac{\epsilon}{2} \left| \tilde{\lambda}_{k+1}^{(t)} - \tilde{\lambda}_k^{(t)} \right| \leq \frac{\epsilon}{2} (|\lambda_{k+1} - \lambda_k| + 2\epsilon_t) \Rightarrow \epsilon_t \leq \epsilon \Delta_k,$$

where in the last we used the assumption that $\epsilon \in (0, 1/2)$. Since at each iteration $\Delta_k - 2\epsilon_t \leq \tilde{\Delta}_k^{(t)}$, then the following condition is sufficient for termination:

$$\epsilon_t \leq \frac{\epsilon}{2} \Delta_k - 2\epsilon_t \Rightarrow \epsilon_t \leq \epsilon \frac{\Delta_k}{6}.$$

This is reached in at most $t = O(\log(\log(\frac{1}{\epsilon \Delta_k})))$ iterations. From Theorem 4.2, each iteration costs

$$O\left(n^\omega \mathcal{F}\left(\log(n) \log(Z\kappa(S)) + \log\left(\frac{1}{\epsilon_t}\right)\right) + n^2 \text{polylog}\left(\frac{n}{\epsilon_t}\right)\right)$$

boolean operations, which is maximized in the last iteration. This gives a total cost of at most

$$O\left(\log(\log(\frac{1}{\epsilon \Delta_k})) \left[n^\omega \mathcal{F}\left(\log(n) \log(Z\kappa(S)) + \log\left(\frac{1}{\epsilon \Delta_k}\right)\right) + n^2 \text{polylog}\left(\frac{n}{\epsilon \Delta_k}\right) \right]\right)$$

boolean operations. We finally return $\tilde{\Delta}_k = \tilde{\lambda}_{k+1}^{(t)} - \tilde{\lambda}_k^{(t)}$ and $\tilde{\mu}_k = \frac{\tilde{\lambda}_{k+1}^{(t)} + \tilde{\lambda}_k^{(t)}}{2}$ which satisfy the advertised bounds. \square

4.4 Spectral projectors and invariant subspaces

For a Hermitian matrix A , the algorithm called GAP in [83] computes the spectral gap and the midpoint in the spirit of Corollary 4.3 using $O\left(n^\omega \log\left(\frac{1}{\epsilon \Delta_k}\right) \log\left(\frac{1}{\delta \epsilon \Delta_k}\right) \cdot \mathcal{F}\left(\log^3\left(\frac{n}{\delta \epsilon \Delta_k}\right) \log\left(\frac{1}{\delta \epsilon \Delta_k}\right) \log(n)\right)\right)$ boolean operations and succeeds with probability $1 - \delta$. On the other hand, if ω is treated as a constant larger than two, the algorithm of Corollary 4.3 requires

$$O\left(n^\omega \mathcal{F}(\log(n)) + n^2 \text{polylog}(n/\epsilon)\right)$$

boolean operations, which is significantly faster than GAP. Moreover, it is deterministic, and it has a lower complexity for banded matrices. A key difference between the two algorithms is that the GAP algorithm of [83] is fully analyzed in floating point, while the algorithm of 4.3 requires [15] which is analyzed in Boolean RAM.

Note that originally the GAP algorithm was used as a preliminary step to locate the gap and the midpoint in order to compute spectral projectors on invariant subspaces. Corollary 4.3 can serve as a direct replacement, providing an end-to-end deterministic algorithm for computing spectral projectors.

Corollary 4.5. *Let (H, S) be a Hermitian definite pencil of size n , with $\|H\|, \|S^{-1}\| \leq 1$, and $\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$ its eigenvalues. Given as input H, S , an integer $1 \leq k \leq n - 1$, an error parameter $\epsilon \in (0, 1)$, we can compute a matrix $\tilde{\Pi}_k$ such that*

$$\left\| \tilde{\Pi}_k - \Pi_k \right\| \leq \epsilon,$$

where Π_k is the true spectral projector on the invariant subspace that is associated with the k smallest eigenvalues, in

$$O\left(n^\omega \mathcal{F}\left(\log(n) \log^3\left(\frac{1}{\Delta_k}\right) \log\left(\frac{n\kappa}{\epsilon \Delta_k}\right)\right) \left(\log\left(\frac{1}{\Delta_k}\right) + \log\left(\log\left(\frac{n\kappa}{\epsilon \Delta_k}\right)\right)\right) + n^2 \text{polylog}\left(\frac{n\kappa}{\Delta_k}\right)\right)$$

boolean operations, where $\kappa := \kappa(S) = \|S\| \|S^{-1}\|$ and $\Delta_k = \lambda_{k+1} - \lambda_k$.

Proof. We first use Corollary 4.1 to compute $\tilde{\kappa} \in [\kappa(S), 3n\kappa(S)]$ in

$$O\left([n^\omega \mathcal{F}(\log(n\kappa)) + n^2 \text{polylog}(n\kappa)] \log(\log(n\kappa))\right)$$

boolean operations. Then, given $\tilde{\kappa}$, we call Corollary 4.4 to compute $\tilde{\mu}_k, \tilde{\Delta}_k, \tilde{\kappa}$ in

$$O\left(\log(\log(\frac{1}{\Delta_k})) \left[n^\omega \mathcal{F}\left(\log(n) \log(n\kappa) + \log(\frac{1}{\Delta_k})\right) + n^2 \text{polylog}(\frac{n}{\Delta_k}) \right]\right)$$

bit operations. Afterwards we can use these values to call [83, Prop. 2.1] to obtain the spectral projector $\tilde{\Pi}_k$ such that $\|\tilde{\Pi}_k - \Pi_k\| \leq \epsilon$. This requires

$$O\left(n^\omega \left(\log(\frac{1}{\Delta_k}) + \log(\log(\frac{n\kappa}{\epsilon \Delta_k}))\right) \cdot \mathcal{F}\left(\log(n) \log^3(\frac{1}{\Delta_k}) \log(\frac{n\kappa}{\epsilon \Delta_k})\right)\right)$$

bit operations. After some simplifications, the total boolean complexity is upper bounded by

$$O\left(n^\omega \mathcal{F}\left(\log(n) \log^3(\frac{1}{\Delta_k}) \log(\frac{n\kappa}{\epsilon \Delta_k})\right) \left[\log(\frac{1}{\Delta_k}) + \log(\log(\frac{n\kappa}{\epsilon \Delta_k}))\right] + n^2 \text{polylog}(\frac{n\kappa}{\Delta_k})\right).$$

□

4.5 Stable inversion and factorizations

Theorem 3.1 can be used to invert a matrix stably in floating point, improving the bit requirement of the logarithmically-stable algorithm of [28], however, at the cost of increasing the arithmetic complexity by a factor of $\log(n)$, which ultimately leads to a slower algorithm. To achieve this, we first form the matrix $\begin{pmatrix} 0 & A^* \\ A & 0 \end{pmatrix}$, and then reduce it to tridiagonal form. Afterwards we can solve $O(n)$ linear systems stably with QR factorization of the tridiagonal matrix, where the right hand sides are the columns of \tilde{Q} , in time $O(n)$ each. We perform one final multiplication to obtain the inverse. We do not expand the analysis further since the boolean complexity of the algorithm is slower than the one of [28] (even if we use a slow algorithm for basic arithmetic operations). The stable inversion algorithm can be used to obtain stable factorization algorithms, such as LU [28] or Cholesky [83].

5 Conclusion

In this work we provided a deterministic complexity analysis for Hermitian eigenproblems. In the Real RAM model, we reported nearly-linear complexity upper bounds for the full diagonalization of arrowhead and tridiagonal matrices, and nearly matrix multiplication-type complexities for diagonalizing Hermitian matrices and for the SVD. This was achieved by analyzing the divide-and-conquer algorithm of [48], when implemented with the Fast Multipole Method. We also showed that the tridiagonal reduction algorithm of [79] is numerically stable in the floating point model. This paved the way to obtain improved deterministic boolean complexities for computing the eigenvalues, singular values, spectral gaps, and spectral projectors, of Hermitian matrices and Hermitian-definite pencils. Some interesting questions for future research are the following.

1. **Stability of arrowhead diagonalization:** The FMM-accelerated arrowhead diagonalization algorithm was only analyzed in the Real RAM model. Several works [48, 87, 70, 21] have provided stabilization techniques of related algorithms in floating point, albeit, without an end-to-end complexity analysis. Such an analysis will be insightful to better understand the boolean complexity of (Hermitian) diagonalization.
2. **Condition number in the SVD complexity:** The complexity of the SVD in Theorem 1.2 has a polylogarithmic dependence on the condition number. Frustratingly, we were not able to remove it at the time of this writing.
3. **Non-Hermitian diagonalization:** Schönhage also proved that a non-Hermitian matrix can be reduced to upper Hessenberg form in matrix multiplication time [79]. In this work we only provided the stability analysis for the Hermitian case. It would be interesting to investigate whether the Hessenberg reduction algorithm can be used to diagonalize non-Hermitian matrices in matrix multiplication time deterministically (e.g. in conjunction with [8, 7, 9]).

4. **Deterministic pseudospectral shattering:** One of the main techniques of the seminal work of [6] is called “pseudospectral shattering.” The main idea is to add a tiny random perturbation to the original matrix to ensure that the minimum eigenvalue gap between any pair of eigenvalues will be at least polynomial in $1/n$. We highlight that the deflation preprocessing step in Proposition A.1 has this precise effect: the pseudospectrum is shattered with respect to a deterministic grid. Can this be generalized to obtain *deterministic* pseudospectral shattering techniques, for Hermitian or even non-Hermitian matrices?

References

- [1] Josh Alman, Ran Duan, Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. More asymmetry yields faster matrix multiplication. In *Proc. 2025 ACM-SIAM Symposium on Discrete Algorithms*, pages 2005–2039. SIAM, 2025.
- [2] Alexandr Andoni and Huy L Nguyen. Eigenvalues of a matrix in the streaming model. In *Proc. 24th ACM-SIAM Symposium on Discrete Algorithms*, pages 1729–1737. SIAM, 2013.
- [3] Diego Armentano, Carlos Beltrán, Peter Bürgisser, Felipe Cucker, and Michael Shub. A stable, polynomial-time algorithm for the eigenpair problem. *Journal of the European Mathematical Society*, 20(6):1375–1437, 2018.
- [4] Grey Ballard, James Demmel, and Nicholas Knight. Communication avoiding successive band reduction. *ACM SIGPLAN Notices*, 47(8):35–44, 2012.
- [5] Grey Ballard, James Demmel, and Nicholas Knight. Avoiding communication in successive band reduction. *ACM Transactions on Parallel Computing*, 1(2):1–37, 2015.
- [6] Jess Banks, Jorge Garza-Vargas, Archit Kulkarni, and Nikhil Srivastava. Pseudospectral Shattering, the Sign Function, and Diagonalization in Nearly Matrix Multiplication Time. *Foundations of Computational Mathematics*, pages 1–89, 2022.
- [7] Jess Banks, Jorge Garza-Vargas, and Nikhil Srivastava. Global Convergence of Hessenberg Shifted QR II: Numerical Stability. *arXiv*, 2022.
- [8] Jess Banks, Jorge Garza-Vargas, and Nikhil Srivastava. Global convergence of Hessenberg Shifted QR I: Exact Arithmetic. *Foundations of Computational Mathematics*, pages 1–34, 2024.
- [9] Jess Banks, Jorge Garza-Vargas, and Nikhil Srivastava. Global Convergence of Hessenberg Shifted QR III: Approximate Ritz Values via Shifted Inverse Iteration. *SIAM Journal on Matrix Analysis and Applications*, 46(2):1212–1246, 2025.
- [10] Jesse L Barlow. Error analysis of update methods for the symmetric eigenvalue problem. *SIAM Journal on Matrix Analysis and Applications*, 14(2):598–618, 1993.
- [11] Michael Ben-Or and Lior Eldar. A Quasi-Random Approach to Matrix Spectral Analysis. In *Proc. 9th Innovations in Theoretical Computer Science Conference*, pages 6:1–6:22. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018.
- [12] Rajendra Bhatia. *Perturbation bounds for matrix eigenvalues*. SIAM, 2007.
- [13] Dario Bini and Victor Pan. Parallel complexity of tridiagonal symmetric eigenvalue problem. In *Proc. 2nd ACM-SIAM Symposium on Discrete Algorithms*, pages 384–393, 1991.
- [14] Dario Bini and Victor Pan. Practical improvement of the divide-and-conquer eigenvalue algorithms. *Computing*, 48(1):109–123, 1992.
- [15] Dario Bini and Victor Y Pan. Computing matrix eigenvalues and polynomial zeros where the output is real. *SIAM Journal on Computing*, 27(4):1099–1115, 1998.

- [16] Christian H Bischof, Bruno Lang, and Xiaobai Sun. A framework for symmetric band reduction. *ACM Transactions on Mathematical Software*, 26(4):581–601, 2000.
- [17] D Boley and Gene Howard Golub. Inverse eigenvalue problems for band matrices. In *Numerical Analysis: Proceedings of the Biennial Conference Held at Dundee, June 28–July 1, 1977*, pages 23–31. Springer, 1977.
- [18] Christos Boutsidis and David P Woodruff. Optimal CUR matrix decompositions. In *Proc. 46th ACM Symposium on Theory of Computing*, pages 353–362, 2014.
- [19] Christos Boutsidis, David P Woodruff, and Peilin Zhong. Optimal principal component analysis in distributed and streaming models. In *Proc. 48th ACM Symposium on Theory of Computing*, pages 236–249, 2016.
- [20] Christos Boutsidis, Anastasios Zouzias, Michael W Mahoney, and Petros Drineas. Randomized dimensionality reduction for k -means clustering. *IEEE Transactions on Information Theory*, 61(2):1045–1062, 2014.
- [21] Difeng Cai and Jianlin Xia. A stable matrix version of the fast multipole method: stabilization strategies and examples. *ETNA-Electronic Transactions on Numerical Analysis*, 54, 2020.
- [22] Kenneth L Clarkson and David P Woodruff. Low-rank approximation and regression in input sparsity time. *Journal of the ACM*, 63(6):1–45, 2017.
- [23] Michael B Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k -means clustering and low rank approximation. In *Proc. 47th ACM Symposium on Theory of Computing*, pages 163–172, 2015.
- [24] Jan JM Cuppen. A divide and conquer method for the symmetric tridiagonal eigenproblem. *Numerische Mathematik*, 36:177–195, 1980.
- [25] Eric Darve. The fast multipole method i: Error analysis and asymptotic complexity. *SIAM Journal on Numerical Analysis*, 38(1):98–128, 2000.
- [26] Eric Darve. The fast multipole method: numerical implementation. *Journal of Computational Physics*, 160(1):195–240, 2000.
- [27] TJ Dekker and JF Traub. The shifted QR algorithm for Hermitian matrices. *Linear Algebra and its Applications*, 4(3):137–154, 1971.
- [28] James Demmel, Ioana Dumitriu, and Olga Holtz. Fast linear algebra is stable. *Numerische Mathematik*, 108(1):59–91, 2007.
- [29] James Demmel, Ioana Dumitriu, Olga Holtz, and Robert Kleinberg. Fast matrix multiplication is stable. *Numerische Mathematik*, 106(2):199–224, 2007.
- [30] James Demmel, Ioana Dumitriu, and Ryan Schneider. Fast and inverse-free algorithms for deflating subspaces. *arXiv*, 2024.
- [31] James Demmel, Ioana Dumitriu, and Ryan Schneider. Generalized Pseudospectral Shattering and Inverse-Free Matrix Pencil Diagonalization. *Foundations of Computational Mathematics*, pages 1–77, 2024.
- [32] James Demmel and Krešimir Veselić. Jacobi’s method is more accurate than QR. *SIAM Journal on Matrix Analysis and Applications*, 13(4):1204–1245, 1992.
- [33] James W Demmel. *Applied numerical linear algebra*. SIAM, 1997.
- [34] Inderjit Singh Dhillon. *A new $O(N^2)$ algorithm for the symmetric tridiagonal eigenvalue/eigenvector problem*. University of California, Berkeley, 1997.
- [35] Jack Dongarra and Francis Sullivan. Guest Editors Introduction to the top 10 algorithms. *Computing in Science & Engineering*, 2(01):22–23, 2000.

- [36] Jack J Dongarra and Danny C Sorensen. A fully parallel algorithm for the symmetric eigenvalue problem. *SIAM Journal on Scientific and Statistical Computing*, 8(2):s139–s154, 1987.
- [37] Petros Drineas, Michael W Mahoney, and Shan Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, 2008.
- [38] Jeff Erickson, Ivor Van Der Hoog, and Tillmann Miltzow. Smoothing the gap between NP and ER. *SIAM Journal on Computing*, 0(0):FOCS20–102–FOCS20–138, 2022.
- [39] John GF Francis. The QR transformation a unitary analogue to the LR transformation—Part 1. *The Computer Journal*, 4(3):265–271, 1961.
- [40] John GF Francis. The QR transformation—Part 2. *The Computer Journal*, 4(4):332–345, 1962.
- [41] Alan Frieze, Ravi Kannan, and Santosh Vempala. Fast Monte-Carlo algorithms for finding low-rank approximations. *Journal of the ACM*, 51(6):1025–1041, 2004.
- [42] Martin Fürer. Faster integer multiplication. In *Proc. 39th ACM Symposium on Theory of Computing*, pages 57–66, 2007.
- [43] Doron Gill and Eitan Tadmor. An $O(N^2)$ Method for Computing the Eigensystem of $N \times N$ Symmetric Tridiagonal Matrices by the Divide and Conquer Approach. *SIAM Journal on Scientific and Statistical Computing*, 11(1):161–173, 1990.
- [44] Gene Golub and William Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis*, 2(2):205–224, 1965.
- [45] Gene H Golub and Henk A Van der Vorst. Eigenvalue computation in the 20th century. *Journal of Computational and Applied Mathematics*, 123(1-2):35–65, 2000.
- [46] Gene H Golub and Charles F Van Loan. *Matrix Computations*. Johns Hopkins University Press, 2013.
- [47] Ming Gu and Stanley C Eisenstat. A stable and fast algorithm for updating the singular value decomposition, 1993.
- [48] Ming Gu and Stanley C Eisenstat. A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem. *SIAM Journal on Matrix Analysis and Applications*, 16(1):172–191, 1995.
- [49] Ming Gu and Stanley C Eisenstat. Efficient algorithms for computing a strong rank-revealing qr factorization. *SIAM Journal on Scientific Computing*, 17(4):848–869, 1996.
- [50] Juris Hartmanis and Janos Simon. On the power of multiplication in random access machines. In *15th Annual Symposium on Switching and Automata Theory*, pages 13–23. IEEE, 1974.
- [51] David Harvey and Joris Van Der Hoeven. Integer multiplication in time $O(n \log n)$. *Annals of Mathematics*, 193(2):563–617, 2021.
- [52] Nicholas J Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002.
- [53] Walter Hoffmann and Beresford N Parlett. A new proof of global convergence for the tridiagonal QL algorithm. *SIAM Journal on Numerical Analysis*, 15(5):929–937, 1978.
- [54] Alston S Householder. Unitary triangularization of a nonsymmetric matrix. *Journal of the ACM*, 5(4):339–342, 1958.
- [55] C.G.J. Jacobi. Über ein leichtes Verfahren die in der Theorie der Säcularstörungen vorkommenden Gleichungen numerisch aufzulösen. *Journal für die reine und angewandte Mathematik*, 30:51–94, 1846.
- [56] Ian T Jolliffe. *Principal component analysis for special types of data*. Springer, 2002.

- [57] Praneeth Kacham and David P Woodruff. Faster Algorithms for Schatten- p Low Rank Approximation. In *Proc. Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024.
- [58] Vera N Kublanovskaya. On some algorithms for the solution of the complete eigenvalue problem. *USSR Computational Mathematics and Mathematical Physics*, 1(3):637–657, 1962.
- [59] Cornelius Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of Research of the National Bureau of Standards*, 45(4), 1950.
- [60] Oren E Livne and Achi Brandt. N roots of the secular equation in $O(N)$ operations. *SIAM Journal on Matrix Analysis and Applications*, 24(2):439–453, 2002.
- [61] Anand Louis and Santosh S Vempala. Accelerated newton iteration for roots of black box polynomials. In *Proc. 57th IEEE Symposium on Foundations of Computer Science*, pages 732–740. IEEE, 2016.
- [62] Per-Gunnar Martinsson and Vladimir Rokhlin. An accelerated kernel-independent fast multipole method in one dimension. *SIAM Journal on Scientific Computing*, 29(3):1160–1178, 2007.
- [63] A Melman. Numerical solution of a secular equation. *Numerische Mathematik*, 69:483–493, 1995.
- [64] RV Mises and Hilda Pollaczek-Geiringer. Praktische Verfahren der Gleichungsauflösung. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 9(1):58–77, 1929.
- [65] Cameron Musco, Christopher Musco, and Aaron Sidford. Stability of the Lanczos method for matrix function approximation. In *Proc. 29th ACM-SIAM Symposium on Discrete Algorithms*, pages 1605–1624. SIAM, 2018.
- [66] Yuji Nakatsukasa, Zhaojun Bai, and François Gygi. Optimizing Halley’s iteration for computing the matrix polar decomposition. *SIAM Journal on Matrix Analysis and Applications*, 31(5):2700–2720, 2010.
- [67] Yuji Nakatsukasa and Nicholas J Higham. Stable and efficient spectral divide and conquer algorithms for the symmetric eigenvalue decomposition and the SVD. *SIAM Journal on Scientific Computing*, 35(3):A1325–A1349, 2013.
- [68] Deanna Needell, William Swartworth, and David P Woodruff. Testing positive semidefiniteness using linear measurements. In *Proc. 2022 IEEE Symposium on Foundations of Computer Science*, pages 87–97. IEEE, 2022.
- [69] Dianne P O’Leary and Gilbert W Stewart. Computing the eigenvalues and eigenvectors of symmetric arrowhead matrices. *Journal of Computational Physics*, 90(2):497–505, 1990.
- [70] Xiaofeng Ou and Jianlin Xia. Superdc: superfast divide-and-conquer eigenvalue decomposition with improved stability for rank-structured matrices. *SIAM Journal on Scientific Computing*, 44(5):A3041–A3066, 2022.
- [71] Chris C Paige. Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem. *Linear algebra and its Applications*, 34:235–258, 1980.
- [72] Victor Y Pan and Zhao Q Chen. The complexity of the matrix eigenproblem. In *Proc. 31st ACM Symposium on Theory of Computing*, pages 507–516, 1999.
- [73] Christos H Papadimitriou, Hisao Tamaki, Prabhakar Raghavan, and Santosh Vempala. Latent semantic indexing: A probabilistic analysis. In *Proc. 17th ACM Symposium on Principles of Database Systems*, pages 159–168, 1998.
- [74] Beresford N Parlett. *The Symmetric Eigenvalue Problem*. SIAM, 1998.
- [75] Vladimir Rokhlin. Rapid solution of integral equations of classical potential theory. *Journal of computational physics*, 60(2):187–207, 1985.

- [76] H Rutishauser. On Jacobi rotation patterns. In *Proceedings of Symposia in Applied Mathematics*, volume 15, pages 219–239, 1963.
- [77] Yousef Saad. *Numerical methods for large eigenvalue problems: revised edition*. SIAM, 2011.
- [78] Ryan Schneider. *Pseudospectral divide-and-conquer for the generalized eigenvalue problem*. University of California, San Diego, 2024.
- [79] Arnold Schönhage. Unitäre transformationen großer matrizen. *Numerische Mathematik*, 20:409–417, 1972.
- [80] Arnold Schönhage. On the power of random access machines. In *Proc. International Colloquium on Automata, Languages, and Programming*, pages 520–529. Springer, 1979.
- [81] Arnold Schönhage and Volker Strassen. Fast multiplication of large numbers. *Computing*, 7:281–292, 1971.
- [82] Rikhav Shah. Hermitian Diagonalization in Linear Precision. In *Proc. 2025 ACM-SIAM Symposium on Discrete Algorithms*, pages 5599–5615. SIAM, 2025.
- [83] Aleksandros Sobczyk, Marko Mladenovic, and Mathieu Luisier. Invariant subspaces and PCA in nearly matrix multiplication time. *Advances in Neural Information Processing Systems*, 37:19013–19086, 2024.
- [84] Nevena Jakovčević Stor, Ivan Slapničar, and Jesse L Barlow. Accurate eigenvalue decomposition of real symmetric arrowhead matrices and applications. *Linear Algebra and its Applications*, 464:62–89, 2015.
- [85] Xiaobai Sun and Nikos P Pitsianis. A matrix version of the fast multipole method. *SIAM Review*, 43(2):289–300, 2001.
- [86] William Swartworth and David P Woodruff. Optimal eigenvalue approximation via sketching. In *Proc. 55th ACM Symposium on Theory of Computing*, pages 145–155, 2023.
- [87] James Vogel, Jianlin Xia, Stephen Cauley, and Venkataramanan Balakrishnan. Superfast divide-and-conquer method and perturbation analysis for structured eigenvalue solutions. *SIAM Journal on Scientific Computing*, 38(3):A1358–A1382, 2016.
- [88] James Hardy Wilkinson. Global convergene of tridiagonal QR algorithm with origin shifts. *Linear Algebra and its Applications*, 1(3):409–420, 1968.

A Preliminaries for symmetric arrowhead diagonalization

This section contains the required preliminaries to describe the algorithm for symmetric arrowhead diagonalization. Our analysis relies on the methodology of [48], but we refer also to [69, 84] for related techniques. We first state the following lemma which describes the desired properties of the input matrix.

Lemma A.1. *Let $\mathbf{H} = \begin{pmatrix} \alpha & \mathbf{z}^\top \\ \mathbf{z} & \mathbf{D} \end{pmatrix}$, $\|\mathbf{H}\| \leq 1$, be an arrowhead matrix in the form of Eq. (3). Suppose that the elements of \mathbf{H} satisfy the following properties, for all $i = 2, \dots, n$:*

$$d_{i+1} - d_i \geq \tau, \quad \text{and} \quad |\mathbf{z}_i| \geq \tau, \quad (5)$$

for some $\tau \in (0, 1)$, where d_i is the i -th element of \mathbf{D} . The eigenvalues λ_i are the roots of the secular equation

$$f(\lambda) = \lambda - \alpha + \sum_{j=2}^n \frac{\mathbf{z}_j^2}{d_j - \lambda}, \quad (6)$$

and they satisfy the following interlacing property:

$$\lambda_1 < d_2 < \lambda_2 < \dots < d_n < \lambda_n. \quad (7)$$

Moreover, it also holds that

$$\begin{aligned} \min \{ |d_i - \lambda_i|, |d_{i+1} - \lambda_i| \} &\geq \frac{\tau^3}{n+1}, \quad \text{for all } i = 2, \dots, n-1, \\ \min \{ |d_2 - \lambda_1|, |d_n - \lambda_n| \} &\geq \frac{\tau^3}{n}, \end{aligned} \quad (8)$$

i.e., there is a well defined gap between the eigenvalues and their boundaries.

Proof. The fact that the eigenvalues are the roots of $f(\lambda)$ and that they satisfy the interlacing property is well-known (see e.g. [69]). We now prove the lower bound, which will be useful for the complexity analysis.

If λ_i is an eigenvalue, for some $i \in \{1, \dots, n\}$, it holds that $f(\lambda_i) = 0$. By expanding and rearranging the equation $f(\lambda_i) = 0$, we can write

$$\frac{\mathbf{z}_i^2}{d_i - \lambda_i} = \alpha - \lambda_i - \sum_{j=2}^{i-1} \frac{\mathbf{z}_j^2}{d_j - \lambda_i} - \sum_{j=i+1}^n \frac{\mathbf{z}_j^2}{d_j - \lambda_i}. \quad (9)$$

For $i = n$, the right hand side of Eq. (9) can be bounded as

$$\left| \alpha - \lambda_n - \sum_{j=2}^{n-1} \frac{\mathbf{z}_j^2}{d_j - \lambda_n} \right| \leq |\alpha| + |\lambda_n| + \sum_{j=2}^{n-1} \frac{\mathbf{z}_j^2}{|d_j - \lambda_n|} \leq 1 + 1 + (n-2) \frac{1}{\tau} \leq \frac{n}{\tau}.$$

Then

$$\frac{\mathbf{z}_n^2}{|d_n - \lambda_n|} \leq \frac{n}{\tau} \Rightarrow |d_n - \lambda_n| \geq \frac{\tau \mathbf{z}_n^2}{n} \geq \frac{\tau^3}{n}.$$

The corresponding bound for $i = 1$ can be obtained with similar arguments after rearranging Eq. (9).

For $i = 2, \dots, n-1$, consider the following three cases:

1. $\lambda_i = \frac{d_{i+1} + d_i}{2}$: In this case λ_i lies exactly in the center of the interval $[d_i, d_{i+1}]$, and since $|d_{i+1} - d_i| \geq \tau$, we have that $\min \{ |d_i - \lambda_i|, |d_{i+1} - \lambda_i| \} \geq \frac{\tau}{2} \gg \frac{\tau^3}{n+1}$.

2. $\lambda_i \in (d_i, \frac{d_{i+1}+d_i}{2})$: If λ_i lies inside the left half of the interval, then clearly $|d_{i+1} - \lambda_i| > \frac{\tau}{2}$. The magnitude of the right hand side of (9) can be bounded from above as follows

$$\begin{aligned} \left| \alpha - \lambda_i - \sum_{j=2}^{i-1} \frac{z_j^2}{d_j - \lambda_i} - \sum_{j=i+1}^n \frac{z_j^2}{d_j - \lambda_i} \right| &\leq |\alpha| + |\lambda_i| + \sum_{j=2, j \neq i}^n \frac{z_j^2}{|d_j - \lambda_i|} \\ &\leq 1 + 1 + (n-3)\frac{1}{\tau} + \frac{1}{\tau/2} \\ &\leq \frac{n+1}{\tau}, \end{aligned}$$

which implies

$$\frac{z_i^2}{|d_i - \lambda_i|} \leq \frac{n+1}{\tau} \Rightarrow |d_i - \lambda_i| \geq \frac{\tau z_i^2}{n+1} \geq \frac{\tau^3}{n+1}.$$

Thus $\min \{|d_i - \lambda_i|, |d_{i+1} - \lambda_i|\} \geq \min \left\{ \frac{\tau}{2}, \frac{\tau^3}{n+1} \right\} \geq \frac{\tau^3}{n+1}$.

3. $\lambda_i \in (\frac{d_{i+1}+d_i}{2}, d_{i+1})$: With similar arguments we obtain that $\min \{|d_i - \lambda_i|, |d_{i+1} - \lambda_i|\} \geq \frac{\tau^3}{n+1}$ holds for this case as well, which concludes the proof. \square

A.1 Deflation

In order to ensure that the given matrix satisfies the requirements of Equation (5) we will use deflation, specifically the methodology of section 4 in [48].

Proposition A.1 (Arrowhead deflation). *Let $\mathbf{H} = \begin{pmatrix} \alpha & \mathbf{z}^\top \\ \mathbf{z} & \mathbf{D} \end{pmatrix}$ be an arrowhead matrix in the form of Eq. (3). There exists an orthogonal matrix \mathbf{G} and a matrix $\tilde{\mathbf{H}} = \begin{pmatrix} \tilde{\mathbf{H}}' & 0 \\ 0 & \tilde{\mathbf{D}} \end{pmatrix}$ such that $\tilde{\mathbf{D}}$ is diagonal and $\tilde{\mathbf{H}}' = \begin{pmatrix} \tilde{\alpha}' & \tilde{\mathbf{z}}'^\top \\ \tilde{\mathbf{z}}' & \tilde{\mathbf{D}}' \end{pmatrix}$ is an arrowhead matrix that satisfies the requirements of Equation (5), specifically:*

$$\tilde{d}'_{j+1} - \tilde{d}'_j \geq \tau, \quad |\tilde{z}'_i| \geq \tau, \quad \text{and} \quad \|\mathbf{H} - \mathbf{G}\tilde{\mathbf{H}}\mathbf{G}^\top\| \leq n\tau.$$

$\tilde{\mathbf{H}}$ can be computed in $O(n \log(n))$ arithmetic operations and comparisons, and the product $\mathbf{G}^\top \mathbf{B}$ for some matrix \mathbf{B} with r columns can be computed in additional $O(nr)$ arithmetic operations on-the-fly.

Proof. This directly follows from the deflation strategy of Section 4.1 in [48]. We expand it's analysis here in three steps.

Step 1: Sorting. First we sort the diagonal elements of $\mathbf{D} = \begin{pmatrix} d_2 & & & \\ & d_3 & & \\ & & \ddots & \\ & & & d_n \end{pmatrix}$ in ascending order, and then

apply a permutation to ensure that

$$d_2 \leq d_3 \leq \dots \leq d_n.$$

Let \mathbf{P}_S be the corresponding permutation matrix. This procedure can be achieved in $O(n \log(n))$ comparisons, and it takes $O(n)$ to reorder $\mathbf{H}_S = \mathbf{P}_S \mathbf{H} \mathbf{P}_S^\top$ and $O(nr)$ to reorder $\mathbf{B}_S = \mathbf{P}_S^\top \mathbf{B}$.

Step 2: Shaft deflation. Next, we set all the elements in \mathbf{z} that have magnitude smaller than τ to zero, by computing $\mathbf{H}_Z = \mathbf{H}_S - \mathbf{E}_Z$, where $\|\mathbf{E}_Z\| \leq (n-1)\tau$, since there are at most $n-1$ elements of \mathbf{z} with magnitude less

than τ that were set to zero. We now choose another permutation matrix \mathbf{P}_Z such that the zeros in \mathbf{z} go to the end. This way we obtain a matrix of the form

$$\mathbf{P}_Z \mathbf{H}_Z \mathbf{P}_Z^\top = \begin{pmatrix} \alpha & \mathbf{z}'^\top & 0 \\ \mathbf{z}' & \mathbf{D}' & 0 \\ 0 & 0 & \mathbf{D}'' \end{pmatrix} = \begin{pmatrix} \mathbf{H}'_Z & 0 \\ 0 & \mathbf{D}'' \end{pmatrix},$$

where \mathbf{D}' and \mathbf{D}'' are diagonal, and \mathbf{D}' remains sorted, while the elements of \mathbf{D}'' are eigenvalues and can be deflated. The permutation costs $O(n)$ to apply on \mathbf{H}_Z and $O(nr)$ to apply on \mathbf{B}_S .

Step 3: Diagonal deflation. We now focus on \mathbf{H}'_Z . First, observe that \mathbf{H}'_Z is in fact a principal submatrix of the original matrix \mathbf{H} , i.e. it contains a subset of rows and columns of \mathbf{H} . From the interlacing property, $\|\mathbf{H}'_Z\| \leq \|\mathbf{H}\| \leq 1$. It remains to ensure that \mathbf{D}'_Z satisfies the requirements of Equation (5). Assume that \mathbf{H}'_Z has size $k \times k$, for some $0 \leq k \leq n-1$. We iterate over $j = k-1, \dots, 2$. If we meet a pair of entries such that $d'_{j+1} - d'_j \leq \tau$, then we apply the deflation technique of Equation (15) of [48]. Specifically, we perform the following update:

$$\mathbf{H}'_Z \leftarrow \mathbf{P}_j (\mathbf{G}_j \mathbf{H}'_Z \mathbf{G}_j^\top - \mathbf{E}_j) \mathbf{P}_j^\top,$$

The matrices \mathbf{P}_j , \mathbf{G}_j , and \mathbf{E}_j are described as follows. Let j be the largest index such that $d'_{j+1} - d'_j \leq \tau$. The matrix \mathbf{G}_j encodes an elementary rotation and it has the same size \mathbf{H}'_Z . They have the following form:

$$\mathbf{G}_j = \begin{pmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & \ddots & & & & \\ & & & c_j & -s_j & & \\ & & & s_j & c_j & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{pmatrix}, \quad \mathbf{H}'_Z = \begin{pmatrix} \alpha & \mathbf{z}'_2 & \dots & \mathbf{z}'_j & \mathbf{z}'_{j+1} & \dots & \mathbf{z}'_k \\ \mathbf{z}'_2 & d'_2 & & & & & \\ \vdots & & \ddots & & & & \\ \mathbf{z}'_j & & & d'_j & & & \\ \mathbf{z}'_{j+1} & & & & d'_{j+1} & & \\ \vdots & & & & & \ddots & \\ \mathbf{z}'_k & & & & & & d'_k \end{pmatrix},$$

$$\mathbf{G}_j \mathbf{H}'_Z \mathbf{G}_j^\top = \begin{pmatrix} \alpha & \mathbf{z}'_2 & \dots & \mathbf{z}''_j & 0 & \dots & \mathbf{z}'_k \\ \mathbf{z}'_2 & d'_2 & & & & & \\ \vdots & & \ddots & & & & \\ \mathbf{z}''_j & & & d''_j & \varepsilon_j & & \\ 0 & & & \varepsilon_j & d''_{j+1} & & \\ \vdots & & & & & \ddots & \\ \mathbf{z}'_k & & & & & & d'_k \end{pmatrix}, \quad \begin{aligned} \mathbf{z}''_j &= \sqrt{\mathbf{z}'_j{}^2 + \mathbf{z}'_{j+1}{}^2}, & s_j &= \mathbf{z}'_j / r_j, \\ c_j &= \mathbf{z}'_{j+1} / r_j, & d''_j &= d'_j c_j^2 + d'_{j+1} s_j^2, \\ d''_{j+1} &= d'_j s_j^2 + d'_{j+1} c_j^2, & \varepsilon_j &= c_j s_j (d'_{j+1} - d'_j). \end{aligned}$$

\mathbf{G}_j sets \mathbf{z}'_{j+1} to zero. Recall also that in \mathbf{H}'_Z for all $i = j+2, \dots, k$ it holds that $d'_i - d'_{i-1} > \tau$, and for all $i = 2, \dots, k$

it holds that $|\mathbf{z}'_i| \geq \tau$ and $d'_i \geq d'_{i-1}$. It is easy to see that $|\varepsilon_j| \leq \tau$. By setting $\mathbf{E}_j = \begin{pmatrix} & & & & \\ & 0 & \varepsilon_j & & \\ & \varepsilon_j & 0 & & \\ & & & \ddots & \end{pmatrix}$, we have that

$\|\mathbf{E}_j\| \leq \tau$ and d''_{j+1} becomes an eigenvalue of the matrix $\mathbf{G}_j \mathbf{H}'_Z \mathbf{G}_j^\top - \mathbf{E}_j$. Therefore, if we simply permute d''_{j+1} to the

bottom-right corner with a permutation matrix \mathbf{P}_j , we obtain a matrix

$$\mathbf{P}_j(\mathbf{G}_j \mathbf{H}'_Z \mathbf{G}_j^\top - \mathbf{E}_j) \mathbf{P}_j^\top = \begin{pmatrix} \alpha & \mathbf{z}'_2 & \dots & \mathbf{z}'_j & \dots & \mathbf{z}'_k & 0 \\ \mathbf{z}'_2 & d'_2 & & & & & \\ \vdots & & \ddots & & & & \\ \mathbf{z}'_j & & & d'_j & & & \\ \vdots & & & & \ddots & & \\ \mathbf{z}'_k & & & & & d'_k & \\ 0 & & & & & & d'_{j+1} \end{pmatrix}$$

For d'_j , recall that we can write $d'_{j+1} = d'_j + \gamma_j$, where $\gamma_j \in [0, \tau]$. Then $d'_j = c_j^2 d'_j + (1 - c_j^2) d'_{j+1} = d'_j + (1 - c_j^2) \gamma_j$. Thus, $d'_j \geq d'_j \geq d'_{j-1}$. Also, by assumption $d'_{j+2} - d'_{j+1} > \tau$, which means that $d'_j = d'_j + (1 - c_j^2) \gamma_j \leq d'_{j+1} + \tau < d'_{j+2}$. Combining these two bounds, we conclude that

$$d'_{j-1} \leq d'_j \leq d'_{j+2}.$$

Therefore, the diagonal part of the top-left $(k-1) \times (k-1)$ principal submatrix remains sorted after the deflation.

After the deflation, if $j = k-1$ we update $j \leftarrow j-1$ and $k \leftarrow k-1$ and apply the same on the $k \times k$ leading principal submatrix of $\mathbf{P}_j(\mathbf{G}_j \mathbf{H}'_Z \mathbf{G}_j^\top - \mathbf{E}_j) \mathbf{P}_j^\top$. Otherwise If the new diagonal elements do not satisfy $d'_{j+1} - d'_j > \tau$, we rotate and deflate once more. Otherwise, if $d'_{j+1} - d'_j > \tau$ or if $j = k$ after the deflation, we decrease $j \leftarrow j-1$ and continue. After every step, either the size of the arrowhead matrix (i.e. k) reduces by 1 due to the deflation, or j decreases by 1. Therefore, after at most $n-1$ deflation steps the procedure will terminate. Each deflation step takes $O(1)$ arithmetic operations, and $O(r)$ operations to apply the corresponding rotation/permutation on \mathbf{B} .

Denoting by $\mathbf{H}_0 := \mathbf{P}_Z \mathbf{H}_Z \mathbf{P}_Z^\top = \mathbf{P}_Z (\mathbf{P}_S \mathbf{H} \mathbf{P}_S^\top - \mathbf{E}_Z) \mathbf{P}_Z^\top$, the final matrix \mathbf{H}_f is given by a sequence of operations

$$\begin{aligned} \mathbf{H}_f &= \mathbf{P}_1 \left(\mathbf{G}_1 \left[\dots \right. \right. \\ &\quad \dots \left[\mathbf{P}_{k-2} (\mathbf{G}_{k-2} [\mathbf{P}_{k-1} (\mathbf{G}_{k-1} \mathbf{H}_0 \mathbf{G}_{k-1}^\top - \mathbf{E}_{k-1}) \mathbf{P}_{k-1}^\top] \mathbf{G}_{k-2}^\top - \mathbf{E}_{k-2}) \mathbf{P}_{k-2}^\top \right] \dots \\ &\quad \left. \left. \dots \right] \mathbf{G}_1^\top - \mathbf{E}_1 \right) \mathbf{P}_1^\top \\ &= \mathbf{P}_1 \mathbf{G}_1 \mathbf{P}_2 \mathbf{G}_2 \dots \mathbf{P}_{k-1} \mathbf{G}_{k-1} \mathbf{H}_0 \mathbf{G}_{k-1}^\top \mathbf{P}_{k-1}^\top \dots \mathbf{G}_2^\top \mathbf{P}_2^\top \mathbf{G}_1^\top \mathbf{P}_1^\top + \mathbf{E}_1 \\ &= \mathbf{P}_1 \mathbf{G}_1 \mathbf{P}_2 \mathbf{G}_2 \dots \mathbf{P}_{k-1} \mathbf{G}_{k-1} \mathbf{P}_Z \mathbf{P}_S \mathbf{H} \mathbf{P}_S^\top \mathbf{P}_Z^\top \mathbf{G}_{k-1}^\top \mathbf{P}_{k-1}^\top \dots \mathbf{G}_2^\top \mathbf{P}_2^\top \mathbf{G}_1^\top \mathbf{P}_1^\top + \mathbf{E}_1 + \mathbf{E}_2 \\ &= \mathbf{G}^\top \mathbf{H} \mathbf{G} + \mathbf{E}_1 + \mathbf{E}_2, \end{aligned}$$

where $\mathbf{G}^\top = \mathbf{P}_1 \mathbf{G}_1 \mathbf{P}_2 \mathbf{G}_2 \dots \mathbf{P}_{k-1} \mathbf{G}_{k-1} \mathbf{P}_Z \mathbf{P}_S$ is orthogonal, the matrix \mathbf{E}_2 is a rotated version of \mathbf{E}_Z and therefore $\|\mathbf{E}_2\| = \|\mathbf{E}_Z\| \leq (n-1)\tau$, and \mathbf{E}_1 is the sum of at most $k-1$ rotated matrices \mathbf{E}_j , in which case $\|\mathbf{E}_1\| \leq (k-1)\tau$.

We ultimately set the matrix to be returned $\tilde{\mathbf{H}} = \mathbf{G}^\top \mathbf{H} \mathbf{G} = \mathbf{H}_f - \mathbf{E}_1 - \mathbf{E}_2$. It holds that

$$\|\mathbf{H} - \tilde{\mathbf{H}} \mathbf{G} \mathbf{G}^\top\| = \|\mathbf{G} \mathbf{E}_1 \mathbf{G}^\top + \mathbf{G} \mathbf{E}_2 \mathbf{G}^\top\| \leq n\tau.$$

The total complexity is as follows. $O(n \log(n))$ comparisons are required for the sorting of the initial \mathbf{D} . Step 2 needs $O(n)$ arithmetic operations to deflate and permute \mathbf{H} , and $O(nr)$ are required to apply the transformations on \mathbf{B} . Step 3 executes at most $n-1$ deflations, where each deflation requires $O(1)$ arithmetic operations and $O(r)$ operations to apply on \mathbf{B} . Therefore the total complexity is $O(n \log(n))$ arithmetic operations and comparisons to obtain $\tilde{\mathbf{H}}$, plus additional $O(nr)$ operations to compute $\mathbf{G}^\top \mathbf{B}$ on-the-fly. \square

A.2 Reconstruction from approximate eigenvalues

If we have access to a set of approximate eigenvalues $\hat{\lambda}_i$ of \mathbf{H} that also satisfy the same interlacing property, then we can construct a matrix $\hat{\mathbf{H}}$ that is close to \mathbf{H} , and $\hat{\lambda}_i$ are the eigenvalues of $\hat{\mathbf{H}}$. This is achieved with the following lemma from [17]. We use its restatement from [48].

Lemma A.2 ([17, 48]). Given a set of n numbers $\widehat{\lambda}_1, \dots, \widehat{\lambda}_n$ and a diagonal matrix $\mathbf{D} = \text{diag}(d_2, \dots, d_n)$ such that

$$\widehat{\lambda}_1 < d_2 < \widehat{\lambda}_2 < \dots < d_n < \widehat{\lambda}_n,$$

there exists a symmetric arrowhead matrix $\widehat{\mathbf{H}} = \begin{pmatrix} \widehat{\alpha} & \widehat{\mathbf{z}}^\top \\ \widehat{\mathbf{z}} & \mathbf{D} \end{pmatrix}$, whose eigenvalues are $\widehat{\lambda}_i$. In this case

$$\begin{aligned} |\widehat{\mathbf{z}}_i| &= \sqrt{(d_i - \widehat{\lambda}_1)(\widehat{\lambda}_n - d_i) \prod_{j=2}^{i-1} \frac{\widehat{\lambda}_j - d_i}{d_j - d_i} \prod_{j=i}^{n-1} \frac{\widehat{\lambda}_j - d_i}{d_{j+1} - d_i}}, \\ \widehat{\alpha} &= \widehat{\lambda}_1 + \sum_{j=2}^n (\widehat{\lambda}_j - d_j), \end{aligned} \tag{10}$$

where the sign of $\widehat{\mathbf{z}}_i$ can be chosen arbitrarily.

We shall use the spectral decomposition of $\widehat{\mathbf{H}}$ as a backward approximate spectral decomposition of \mathbf{H} . We write

$$\mathbf{H} = \begin{pmatrix} \alpha & \mathbf{z}^\top \\ \mathbf{z} & \mathbf{D} \end{pmatrix}, \quad \text{and} \quad \widehat{\mathbf{H}} = \begin{pmatrix} \widehat{\alpha} & \widehat{\mathbf{z}}^\top \\ \widehat{\mathbf{z}} & \mathbf{D} \end{pmatrix},$$

in which case $\|\mathbf{H} - \widehat{\mathbf{H}}\| \leq |\alpha - \widehat{\alpha}| + \|\mathbf{z} - \widehat{\mathbf{z}}\|$. It suffices to show that $\widehat{\alpha} \approx \alpha$ and $\widehat{\mathbf{z}} \approx \mathbf{z}$.

Lemma A.3. Let $\mathbf{H} = \begin{pmatrix} \alpha & \mathbf{z}^\top \\ \mathbf{z} & \mathbf{D} \end{pmatrix}$ be a symmetric arrowhead matrix with $\|\mathbf{H}\| \leq 1$, satisfying the properties of Lemma A.1 with parameter τ , and let $\widehat{\lambda}_1, \widehat{\lambda}_2, \dots, \widehat{\lambda}_n$ be a set of approximate eigenvalues that satisfy

$$\widehat{\lambda}_1 < d_2 < \widehat{\lambda}_2 < \dots < d_n < \widehat{\lambda}_n, \quad \text{and} \quad |\widehat{\lambda}_i - \lambda_i| \leq \epsilon \frac{\tau^3}{2(n+1)},$$

for some $\epsilon \in (0, 1/n)$. Then the quantities $\widehat{\alpha}$, $\widehat{\mathbf{z}}$, and $\widehat{\mathbf{H}}$ from Lemma A.2 satisfy:

$$|\alpha - \widehat{\alpha}| \leq \frac{\epsilon \tau^3}{2}, \quad \|\mathbf{z} - \widehat{\mathbf{z}}\| \leq \frac{n\epsilon}{1 - n\epsilon}, \quad \|\mathbf{H} - \widehat{\mathbf{H}}\| \leq \frac{\epsilon \tau^3}{2} + \frac{n\epsilon}{1 - n\epsilon}.$$

Proof. From Eq. (10) we have that $\alpha = \lambda_1 + \sum_{j=2}^n (\lambda_j - d_j)$ and $\widehat{\alpha} = \widehat{\lambda}_1 + \sum_{j=2}^n (\widehat{\lambda}_j - d_j)$. Then:

$$|\alpha - \widehat{\alpha}| = \left| \sum_{j=1}^n (\lambda_j - \widehat{\lambda}_j) \right| \leq \sum_{j=1}^n |\lambda_j - \widehat{\lambda}_j| \leq \epsilon \frac{\tau^3}{2(n+1)} n \leq \frac{\epsilon \tau^3}{2}.$$

We know that $|\widehat{\lambda}_i - \lambda_i| \leq \epsilon \frac{\tau^3}{2(n+1)}$. Combined with Lemma A.1, it implies that

$$|\lambda_i - \widehat{\lambda}_i| \leq \epsilon |\lambda_i - d_i|, \quad \text{for all } i = 2, \dots, n,$$

$$|\lambda_i - \widehat{\lambda}_i| \leq \epsilon |\lambda_i - d_{i+1}|, \quad \text{for all } i = 1, \dots, n-1.$$

Since $\max\{|\lambda_i - d_i|, |\lambda_i - d_{i+1}|\} \leq |\lambda_i - d_j|$, for all $j = 1, \dots, i-1, i+2, \dots, n$, then we can write

$$\widehat{\lambda}_i - d_j = \widehat{\lambda}_i - \lambda_i + \lambda_i - d_j = (1 + \epsilon_{ij})(\lambda_i - d_j),$$

for all $i, j \in [n]$, where $|\epsilon_{ij}| \leq \epsilon$. Then from Equation (10) the elements of $\widehat{\mathbf{z}}$ are

$$\begin{aligned} |\widehat{\mathbf{z}}_i| &= \sqrt{(d_i - \widehat{\lambda}_1)(\widehat{\lambda}_n - d_i) \prod_{j=2}^{i-1} \frac{\widehat{\lambda}_j - d_i}{d_j - d_i} \prod_{j=i}^{n-1} \frac{\widehat{\lambda}_j - d_i}{d_{j+1} - d_i}} \\ &= \sqrt{(d_i - \lambda_1)(1 + \epsilon_{1i})(\lambda_n - d_i)(1 + \epsilon_{ni}) \prod_{j=2}^{i-1} \frac{\lambda_j - d_i}{d_j - d_i} (1 + \epsilon_{ji}) \prod_{j=i}^{n-1} \frac{\lambda_j - d_i}{d_{j+1} - d_i} (1 + \epsilon_{ji})} \\ &= |\mathbf{z}_i| \sqrt{\prod_{j=1}^n (1 + \epsilon_{ji})}. \end{aligned}$$

By assumption, $\epsilon < 1/n$. This allows us to use standard rounding error analysis (see e.g. [52]), from which we know that $\prod_{j=1}^n (1 + \epsilon_{ij}) = 1 + \theta_i$, where $|\theta_i| \leq \frac{n\epsilon}{1-n\epsilon}$.

Finally, recall that from Lemma A.2, we can choose the sign of $\widehat{\mathbf{z}}_i$ arbitrarily. By choosing the sign of $\widehat{\mathbf{z}}_i$ to be the same as \mathbf{z}_i , we finally obtain

$$\begin{aligned} |\mathbf{z}_i - \widehat{\mathbf{z}}_i| &= |\mathbf{z}_i| \left| \sqrt{1 + \theta_i} - 1 \right| = |\mathbf{z}_i| \left| \frac{\theta_i}{\sqrt{1 + \theta_i} + 1} \right| \leq |\mathbf{z}_i| |\theta_i| \leq |\mathbf{z}_i| \frac{n\epsilon}{1-n\epsilon}, \\ \|\mathbf{z} - \widehat{\mathbf{z}}\| &\leq \frac{n\epsilon}{1-n\epsilon} \|\mathbf{z}\| \leq \frac{n\epsilon}{1-n\epsilon}. \end{aligned}$$

□

B Fast Arrowhead diagonalization

In this section we provide the full analysis of the algorithm of [48] when accelerated with the FMM. The FMM was introduced in [75], to accelerate the evaluation of integrals between interacting bodies, and it has been comprehensively analyzed in the literature [26, 25, 21, 85, 62, 60, 47]. Consider a function of the form

$$f(x) = \sum_{j=1}^n c_j k(x - x_j), \quad (11)$$

where x_j, c_j are constants and $k(x)$ is a suitably chosen kernel function, typically one of $\{\log(x), \frac{1}{x}, \frac{1}{x^2}\}$. The FMM can be used to approximately evaluate $f(x)$ over m different points x_i in only $\tilde{O}(m+n)$ arithmetic operations (suppressing logarithmic terms in the accuracy), instead of the naive $O(mn)$ evaluation.

B.1 Fast Multipole Method

For our analysis, we will need to use the FMM to evaluate the following functions (12)-(16), each on $O(n)$ points. For each function, there is a guarantee that the evaluation points will satisfy certain criteria. More precisely, the magnitude of the denominators and the logarithm arguments will have a well-defined lower bound. Here $\tau \in (0, 1)$ is a parameter that is determined later.

Function:	Guarantees for FMM:	Used in:
$f(\lambda) = \sum_{j=1}^n \frac{\mathbf{z}_j^2}{d_j - \lambda},$	$ d_j - \lambda \geq \Omega(\text{poly}(\frac{\tau}{n})),$	Lemma B.1, (12)

$f(d_i) = \sum_{j=2}^n \log(\widehat{\lambda}_j - d_i),$	$ \widehat{\lambda}_j - d_i \geq \Omega(\text{poly}(\frac{\tau}{n})),$	Lemma B.2, (13)
--	---	-----------------

$f(d_i) = \sum_{j=2}^n \log(d_j - d_i),$	$ d_j - d_i \geq \tau,$	Lemma B.2, (14)
--	--------------------------	-----------------

$f(\lambda) = \sum_{k=2}^n \frac{(1 + \epsilon_k) \widehat{\mathbf{z}}_k \mathbf{q}_k}{d_k - \lambda},$	$ d_k - \lambda \geq \Omega(\text{poly}(\frac{\tau}{n})),$	Lemma B.3, (15)
---	---	-----------------

$f(\lambda) = \sum_{k=2}^n \frac{(1 + \epsilon_k)^2 \widehat{\mathbf{z}}_k^2}{(d_k - \lambda)^2},$	$ d_k - \lambda \geq \Omega(\text{poly}(\frac{\tau}{n})),$	Lemma B.3. (16)
--	---	-----------------

The fact that the magnitudes of the denonimators and the logarithm arguments are bounded from below allows us to use the seminal FMM analysis of [47, 60, 21]. To the best of our knowledge, [47] is one of the first works to rigorously analyze the application of the FMM on such kernel functions with end-to-end approximation and complexity bounds. In Section 3.4 they achieved an error of $O\left(\epsilon \sum_i \frac{|x_i|}{|\omega^2 - d_i^2|}\right)$ for the function $\Phi(\omega) = \sum_i \frac{x_i}{d_i^2 - \omega^2}$,

assuming that d_i and ω satisfy similar interlacing properties to ours, in $O(n \log^2(1/\epsilon))$ arithmetic operations. This complexity translates to $O(n \log^2(\frac{n}{\epsilon}))$ arithmetic operations, if we rescale ϵ appropriately to obtain an absolute error $O(\epsilon)$, i.e., it satisfies the guarantees of Proposition 1.1 with $\xi = 2$. [60] used a kernel-softening approach and report similar bounds in $O(n \log(1/\epsilon))$ operations. More recently, [21] developed a general framework based on the matrix-version of the FMM [85], and provided a thorough analysis and explicit bounds similar to [47], which can be used to obtain guarantees for all the functions (12)-(16).

For completeness, we summarize the main ideas and results of the aforementioned works and provide a short proof for Proposition 1.1 below. We remind once more that we do not account for floating point errors, but rather focus only on the error FMM approximation errors. Obtaining full, end-to-end complexity analysis under floating point errors has its own merit, and it is left as future work.

Proposition B.1 (FMM). *There exists an algorithm, which we refer to as (ϵ, n) -approximate FMM (or (ϵ, n) -FMM, for short), which takes as input*

- a kernel function $k(x) = \{\log(|x|), \frac{1}{x}, \frac{1}{x^2}\}$,
- $2n + m$ real numbers: $\{x_1, \dots, x_m\} \cup \{c_1, \dots, c_n\} \cup \{y_1, \dots, y_n\}$, and a constant C , such that $m \leq n$ and for all $i \in [m], j \in [n]$ it holds that

$$|x_i|, |c_j|, |y_j| < C \quad \text{and} \quad |x_i - y_j| \geq \Omega(\text{poly}(\frac{\epsilon}{n})).$$

It returns m values $\tilde{f}(x_1), \dots, \tilde{f}(x_m)$ such that $|\tilde{f}(x_i) - f(x_i)| \leq \epsilon$, for all $i \in [m]$, where $f(x) = \sum_{j=1}^n c_j k(x_i - y_j)$, in a total of $O\left(n \log^\xi(\frac{n}{\epsilon})\right)$ arithmetic operations, where $\xi \geq 1$ is a small constant that is independent of ϵ, n .

Proof. All the evaluation points are always confined inside the interval $[-1, 1]$.

Following the seminal analysis of [21, Sec. 2.1], and simplifying it for the real case in exact arithmetic, let us initially focus on the kernel function $k(x - y) = \frac{1}{x - y}$. Consider two sets of points: $X = \{x_1, \dots, x_m\}$ and $Y = \{y_1, \dots, y_n\}$, where $x_i, y_j \in [-1, 1]$. Moreover, assume that X and Y are β -separated, with separation ratio β , which means that

$$\rho_X + \rho_Y \leq \beta |\mu_X - \mu_Y|, \quad \beta \in (0, 1),$$

where $\mu_X = \frac{\sum x_i}{m}$ and $\mu_Y = \frac{\sum y_i}{n}$ are the centers of X and Y , while $\rho_X = \max |x_i - \mu_X|$ and $\rho_Y = |y_i - \mu_Y|$ are the radii. Using a Taylor expansion we can write $k(x - y)$ in the so-called matrix form ([85]) as follows:

$$k(x - y) = \mathbf{u}^\top \bar{\mathbf{B}} \mathbf{v} + \epsilon_r, \tag{17}$$

where

$$\begin{aligned} \mathbf{u} &= (f_0(x - \mu_X) \quad f_1(x - \mu_X) \quad \dots \quad f_{r-1}(x - \mu_X))^\top \\ \mathbf{v} &= (f_0(y - \mu_Y) \quad f_1(y - \mu_Y) \quad \dots \quad f_{r-1}(y - \mu_Y))^\top \\ \bar{\mathbf{B}} &= \begin{pmatrix} a_0 & a_1 & \dots & a_{r-1} \\ a_1 & \ddots & & \\ \vdots & \ddots & & \\ a_{r-1} & & & 0 \end{pmatrix} \begin{pmatrix} (-1)^0 & & & \\ & (-1)^1 & & \\ & & \ddots & \\ & & & (-1)^{r-1} \end{pmatrix}, \\ f_j(x) &= \frac{x^j}{j!}, \quad a_j = -\frac{j!}{(\mu_Y - \mu_X)^{j+1}}, \quad |\epsilon_r| \leq \beta^r \frac{1 + \beta}{1 - \beta} |k(x - y)|, \end{aligned}$$

where the error bound holds due to the β -separation.

Adapting the formulation for all $x \in X$ and $y \in Y$, we can write the entire interaction matrix as

$$\mathbf{K} = \bar{\mathbf{U}} \bar{\mathbf{B}} \bar{\mathbf{V}}^\top + \mathbf{K} \odot \mathbf{E},$$

where \odot is the element-wise product, $\mathbf{K}_{i,j} = |k(x_i - y_j)|$, and $|\mathbf{E}_{i,j}| \leq \beta^r \frac{1+\beta}{1-\beta}$.

The goal of FMM is to efficiently approximate the matrix-vector product $\mathbf{K}\mathbf{e}$, where \mathbf{e} is the all-ones vector. This is achieved by truncating the $\mathbf{K} \odot \mathbf{E}$ term. In this case the total error for each evaluation point x_i is bounded by

$$\epsilon_i := |\mathbf{e}_i^\top (\mathbf{K} \odot \mathbf{E}) \mathbf{e}| \leq \beta^r \frac{1+\beta}{1-\beta} \sum_{j=1}^n |k(x_i - y_j)|. \quad (18)$$

Assuming that the matrices have been pre-computed in $O((m+n) \text{poly}(r))$ arithmetic operations, the product $\bar{\mathbf{U}}\bar{\mathbf{B}}\bar{\mathbf{V}}^\top \mathbf{e}$ takes $O(r(r+m+n))$ operations.

Next, we partition $[-1, 1]$ in a constant number of subintervals with equal size, e.g., $I_1 = [-1, -1/2]$, $I_2 = [-1/2, 0]$, $I_3 = [0, 1/2]$, and $I_4 = [1/2, 1]$. Note that the sets I_1 and I_3 are $\frac{1}{2}$ -separated. The same holds for (I_2, I_4) , while (I_1, I_4) are $\frac{1}{3}$ -separated. We can execute the following steps:

1. Split the sets X and Y into four subsets X_1, X_2, X_3, X_4 , and Y_1, Y_2, Y_3, Y_4 , one subset for each interval I_1, I_2, I_3, I_4 .
2. Use a truncated Taylor expansion for each combination of “long-range” interactions, i.e., for the set pairs $(X_1, Y_3), (X_1, Y_4), (X_2, Y_4), (X_3, Y_1), (X_4, Y_1)$, and (X_4, Y_2) .
3. For the remaining ten pairs of “short-range” interactions, apply these steps recursively.
4. Stop when there are no more short-range interactions, i.e., when all the remaining evaluations are between sets that are at least $\frac{1}{2}$ -separated.

Due to the deflation process, we have a guarantee that for *all* $x \in X, y \in Y$, it holds that $|x - y| \geq \theta$, for some $\theta \in (0, 1)$ to be specified later. Let $I_j^{(t)}$, $j = 1, 2, 3, 4$, be the four intervals at recursion depth $t \geq 1$. We know that $|I_j^{(t)}| = 2 \cdot 4^{-t}$. Due to the lower bound on $|x - y|$, the final recursion depth is at most $t = O(\log(1/\theta))$, since at this depth every interval contains at most one element, either from X or from Y . Moreover, all non-empty intervals are at least $\frac{1}{2}$ -separated.

We now calculate the total number of arithmetic operations, starting at the bottom of the recursion tree, i.e., $t = C \log(1/\theta)$ for some constant C . The size of each interval at this depth is $2 \cdot 4^{-C \log(1/\theta)} = 2 \cdot \theta^{2C}$. Therefore, there are a total of $\frac{|[-1, 1]|}{2 \cdot \theta^{2C}} = \frac{1}{\theta^{2C}}$ intervals, bundled in groups of four consecutive intervals. Of all those intervals, only $m + n$ are non-empty. Moreover, inside each group of four, all non-empty ones are at least $\frac{1}{2}$ -separated, which implies that there are at most 2 non-empty intervals in each group of four, each one containing a single element. It is therefore sufficient to execute $O(m + n)$ evaluations using Eq. (17) with a single vector each ($O(1)$ evaluations within each group of four). Each evaluation costs $O(r^2)$, which is independent of m and n . Therefore, the total cost to calculate “short-range” interactions at the bottom of the recursion tree is at most $O((m + n)r^2)$.

Next, we upper bound the cost for the recursion depth $t - 1$, given the cost of depth t . Level t prepares all the short-range interactions of level $t - 1$, in time $T(t - 1)$. At level $t - 1$, there are four times fewer “groups” of intervals, and the intervals have four times larger size. Since the short-range interactions within each group were already prepared, it suffices to calculate the long-range interactions. We use again Eq. (17), but now potentially with more than one vectors within each group. However, the total time for this calculation within each group scales linearly to the total number of points X and Y *within each interval*. In particular, the cost within each (non-empty) group l is $O(r^2(m_l + n_l))$. Summing the cost in all non-empty groups together, the total time to calculate the long-range interactions in *all* groups scales as $O(r^2 \sum_l (m_l + n_l)) = O(r^2(m + n))$. Since there are $O(\log(1/\theta))$ recursive steps, the total cost is bounded by $O(r^2(m + n) \log(1/\theta))$.

It remains to bound the errors and the value of r . The final result for each evaluation point x_i is a sum of applications of Eq. (17), and therefore the bound of Eq. (18) applies for the total error, with $\beta \leq 1/2$. Due to the deflation, we have that $|k(x_i - y_j)| \leq \frac{1}{\theta}$, which means that Eq. (18) becomes

$$\epsilon_i \leq 3 \left(\frac{1}{2}\right)^r \binom{n}{2} \frac{1}{\theta}.$$

If we set $r = O(\log(\frac{n}{\epsilon\theta}))$, then it holds that $\epsilon_i \leq \epsilon$, for all $i \in [n]$. Using the assumption of Proposition 1.1 that $\theta \geq \Omega(\text{poly}(\frac{\epsilon}{n}))$, we obtain the final bound for $r = O(\log(n/\epsilon))$.

In a similar way we can analyze the other kernel functions of interest, namely, $\frac{1}{(x-y)^2}$ and $\log(|x-y|)$. Using their Taylor expansions we can write

$$\frac{1}{(x-y)^2} = \frac{1}{(\rho_X - \rho_Y)^2} \sum_{j=0}^r j \frac{[(x - \rho_X) - (y - \rho_Y)]^{j-1}}{(\rho_Y - \rho_X)^{j-1}} + \eta_r,$$

$$\log(|x-y|) = \log(|\rho_X - \rho_Y|) - \sum_{j=1}^r \frac{|(x - \rho_X) - (y - \rho_Y)|^j}{j|\rho_X - \rho_Y|^j} + \zeta_r.$$

Further expanding the terms in the sums we can arrive to matrix formulations similar to Eq. (17). Moreover, assuming that the underlying sets X and Y are β -separated, the error terms η_r and ζ_r are bounded in a similar way as ϵ_r . With further calculations we can obtain final bounds for r to achieve a total additive error of at most ϵ , similar to the kernel $1/(x-y)$.

The analysis can be straightforwardly generalized when the summation coefficients of the kernel function are different than one, but upper-bounded by some value C . Indeed, it suffices to replace the all-ones vector \mathbf{e} in Eq. (18) with a vector that contains the numerators. If the upper bound C is a constant the analysis is not affected. \square

B.2 Computing the eigenvalues with bisection

Using the FMM as a black-box evaluation of the targeted kernel functions, we can compute all the eigenvalues of an arrowhead matrix which are given by the roots of the secular equation. We highlight that [60] provided a rigorous analysis of the Newton iteration, based on the results of [63], to compute all the roots of the secular equation. The reported complexity is $O(n \log(1/\epsilon))$, however, this does not include the number of Newton steps. Due to the well-known quadratic convergence properties, the number of Newton steps should be bounded by $O(\log(\log(1/\epsilon)))$. For the sake of simplicity and completeness, instead of repeating the analysis of [60], we will use the following standard bisection method to compute all the eigenvalues in a total of $O(n \log^2(1/\epsilon))$ operations. It might be slightly slower than the Newton iteration (up to a log factor), but it is significantly simpler and it does not require the evaluation of derivatives. It should therefore be both easier to implement and to analyze in finite precision.

Lemma B.1. *Let $\mathbf{H} = \begin{pmatrix} \alpha & \mathbf{z}^\top \\ \mathbf{z} & \mathbf{D} \end{pmatrix}$ be a symmetric arrowhead matrix that satisfies the properties of Lemma A.1 for some parameter $\tau \in (0, 1)$, with $\|\mathbf{H}\| \leq 1$, and assume that the diagonal elements of \mathbf{D} are sorted: $d_2 < d_3 < \dots < d_n$. Given $\epsilon \in (0, 1)$, and assuming an (ϵ, n) -FMM implementation as in Proposition 1.1, we can compute approximate eigenvalues $\hat{\lambda}_i$ such that*

$$\hat{\lambda}_1 < d_2 < \hat{\lambda}_2 < \dots < d_n < \hat{\lambda}_n, \quad \text{and} \quad |\hat{\lambda}_i - \lambda_i| \leq \epsilon,$$

in $O\left(n \log(\frac{1}{\epsilon}) \log^\xi(\frac{n}{\tau\epsilon})\right)$ arithmetic operations.

Proof. From Lemma A.1, the eigenvalues λ_i of \mathbf{H} are the roots of the secular equation $f(\lambda) = \lambda - \alpha + \sum_{j=2}^n \frac{z_j^2}{d_j - \lambda}$ and they satisfy the interlacing property

$$-1 \leq \lambda_1 < d_2 < \lambda_2 < d_3 < \dots < d_n < \lambda_n \leq 1,$$

and also that the lower bounds of Equation (8) hold. We thus need to search for the roots of $f(\lambda)$ inside the intervals $I_1 = [-1, d_1 - \frac{\tau^3}{n+1}]$, $I_n = [d_n + \frac{\tau^3}{n+1}, 1]$, and, for all $i = 1, \dots, n-1$, $I_i = [d_i + \frac{\tau^3}{n+1}, d_{i+1} - \frac{\tau^3}{n+1}]$.

Let l_i and r_i be the corresponding left and right boundaries of I_i , and initialize $\epsilon_{FMM} = \epsilon/2$, and $k = 0$ (an iteration counter). A binary search follows.

1. Set $\lambda_i^{(k)} = \frac{l_i + r_i}{2}$ and use the (ϵ_{FMM}, ξ, n) -FMM to compute $\tilde{f}(\lambda_i^{(k)})$ such that $|f(\lambda_i^{(k)}) - \tilde{f}(\lambda_i^{(k)})| \leq \epsilon_{FMM}$.

2. If $|l_i - r_i| \leq \epsilon$, stop and return $\widehat{\lambda}_i = \lambda_i^{(k)}$. (At every step $\lambda_i \in [l, r]$, and therefore $|\lambda_i^{(k)} - \lambda_i| \leq |l_i - r_i| \leq \epsilon$, which means that we can terminate).
3. If $|\widetilde{f}(\lambda_i^{(k)})| \leq \epsilon_{FMM}$, stop and return $\widehat{\lambda}_i = \lambda_i^{(k)}$. In that case we can no longer determine the true sign of $f(\lambda_i^{(k)})$, and thus the binary search cannot continue further.
4. If none of the above criteria are met, then we update the bounding interval $[l_i, r_i]$ as follows:
 - If $\widetilde{f}(\lambda_i^{(k)}) - \epsilon_{FMM} > 0$, we set $r_i = \lambda_i^{(k)}$.
 - Else, if $\widetilde{f}(\lambda_i^{(k)}) + \epsilon_{FMM} < 0$, we set $l_i = \lambda_i^{(k)}$.
5. Update $k \leftarrow k + 1$, and go to Step 1.

If the algorithm stops and returns at Step 2, then $|\lambda_i - \widehat{\lambda}_i| \leq \epsilon$ is already satisfied. On the other hand, if the algorithm stops at Step 3, the returned solution $\widehat{\lambda}_i$ satisfies

$$|f(\widehat{\lambda}_i)| \leq |f(\widehat{\lambda}_i) - \widetilde{f}(\widehat{\lambda}_i)| + |\widetilde{f}(\widehat{\lambda}_i)| \leq \epsilon_{FMM} + \epsilon_{FMM} \leq 2\epsilon_{FMM} = \epsilon.$$

Since $f(\lambda_i) = 0$, we have that

$$\begin{aligned} f(\widehat{\lambda}_i) &= f(\widehat{\lambda}_i) - f(\lambda_i) \\ &= \widehat{\lambda}_i - \alpha + \sum_{j=2}^n \frac{z_j^2}{d_j - \widehat{\lambda}_i} - \lambda_i + \alpha - \sum_{j=2}^n \frac{z_j^2}{d_j - \lambda_i} \\ &= (\widehat{\lambda}_i - \lambda_i) \left[1 + \sum_{j=2}^n \frac{z_j^2}{(d_j - \widehat{\lambda}_i)(d_j - \lambda_i)} \right]. \end{aligned}$$

But $\widehat{\lambda}_i$ and λ_i lie strictly inside the same interval $I_i \subset (d_{i-1}, d_i)$, and therefore the denominator in the sum is always positive, which implies that

$$|\widehat{\lambda}_i - \lambda_i| \leq |\widehat{\lambda}_i - \lambda_i| \left| 1 + \sum_{j=2}^n \frac{z_j^2}{(d_j - \widehat{\lambda}_i)(d_j - \lambda_i)} \right| = |f(\widehat{\lambda}_i) - f(\lambda_i)| = |f(\widehat{\lambda}_i)| \leq \epsilon. \quad (19)$$

Therefore, the returned solution satisfies the desired bound $|\lambda_i - \widehat{\lambda}_i| \leq \epsilon$ regardless of which stopping criterion is met first. The solutions $\widehat{\lambda}_i$ are inside the desired intervals I_i , since we only evaluate the secular equation on points that lie inside I_i . Therefore, the lower bounds of Equation (8) also hold for $\widehat{\lambda}_i$.

At each iteration k , use the FMM to compute $\widetilde{f}(\widehat{\lambda}_i)$ for all i simultaneously using $O(n \log^\xi(\frac{n}{\tau \epsilon_{FMM}})) = O(n \log^\xi(\frac{n}{\tau \epsilon}))$ arithmetic operations. We then need to iterate over all the returned values in $O(n)$ arithmetic operations to determine whether the corresponding eigenvalues have converged or if we need to perform additional iterations.

It remains to bound the number of binary search iterations to reach any of the two termination criteria. Note that at every iteration the algorithm is halving the range of $[l_i, r_i]$ for every eigenvalue. This means that after m iterations λ_i is restricted inside an interval with size at most 2^{-m} . Therefore, the termination criterion of Step 1 is achieved after at most $m = O(\log(1/\epsilon))$ iterations. Since the additional stopping criterion can only make the algorithm terminate faster, the total number of binary search iterations is at most $O(\log(1/\epsilon))$. The total runtime of the algorithm is therefore $O\left(n \log(\frac{1}{\epsilon}) \log^\xi(\frac{n}{\tau \epsilon})\right)$. \square

B.3 Approximating the elements of the shaft

As a next step, we use the trick of [48] to approximate the elements of $\widehat{\mathbf{H}}$ from Lemma A.3.

Lemma B.2. Let $\mathbf{H} = \begin{pmatrix} \alpha & \mathbf{z}^\top \\ \mathbf{z} & \mathbf{D} \end{pmatrix}$ be a symmetric arrowhead matrix with $\|\mathbf{H}\| \leq 1$, that satisfies the requirements of Lemma A.1 with parameter τ , and let $d_2 < \dots < d_n$ be the diagonal elements of \mathbf{D} . From Lemma A.3, if we are given a set of approximate eigenvalues $\hat{\lambda}_i$ that satisfy

$$\hat{\lambda}_1 < d_2 < \hat{\lambda}_2 < \dots < d_n < \hat{\lambda}_n, \quad \text{and} \quad |\hat{\lambda}_i - \lambda_i| \leq \epsilon \frac{\tau^3}{2(n+1)},$$

for some $\epsilon \in (0, 1/n)$, then there exists a matrix $\hat{\mathbf{H}} = \begin{pmatrix} \hat{\alpha} & \hat{\mathbf{z}}^\top \\ \hat{\mathbf{z}} & \mathbf{D} \end{pmatrix}$, such that $\hat{\lambda}_i$ are the exact eigenvalues of $\hat{\mathbf{H}}$ and $\|\mathbf{H} - \hat{\mathbf{H}}\| \leq \frac{\epsilon \tau^3}{2} + \frac{n\epsilon}{1-n\epsilon}$. Assuming an (ϵ, n) -FMM, for any $\epsilon_z \in (0, 1/2)$, we can compute an approximate vector $\hat{\mathbf{z}}'$ such that $|\hat{\mathbf{z}}_i - \hat{\mathbf{z}}'_i| \leq \epsilon_z |\hat{\mathbf{z}}_i|$ and $\|\hat{\mathbf{z}}' - \hat{\mathbf{z}}\| \leq \epsilon_z (1 + \frac{n\epsilon}{1-n\epsilon})$ in $O\left(n \log(\frac{1}{\epsilon_z}) \log^\xi(\frac{n}{\tau \epsilon_z})\right)$ arithmetic operations. The matrix $\hat{\mathbf{H}}' = \begin{pmatrix} \hat{\alpha} & \hat{\mathbf{z}}'^\top \\ \hat{\mathbf{z}}' & \mathbf{D} \end{pmatrix}$ satisfies

$$\|\mathbf{H} - \hat{\mathbf{H}}'\| \leq \frac{\epsilon \tau^3}{2} + \epsilon_z + (1 + \epsilon_z) \frac{n\epsilon}{1-n\epsilon}.$$

Proof. The observation is that $|\hat{\mathbf{z}}_i|$ from Eq. (10) can be written as

$$|\hat{\mathbf{z}}_i| = \sqrt{(d_i - \hat{\lambda}_1)(\hat{\lambda}_n - d_i)} \exp(\Phi(d_i)),$$

where

$$\Phi(d_i) = \frac{1}{2} \left(\sum_{j=2}^{i-1} \log(d_i - \hat{\lambda}_j) - \sum_{j=2}^{i-1} \log(d_i - d_j) + \sum_{j=i+1}^n \log(\hat{\lambda}_j - d_i) - \sum_{j=i+1}^n \log(d_j - d_i) \right).$$

We thus need to evaluate the functions (13)-(16), each at n points, where all the points satisfy the corresponding guarantees. We can thus choose $\epsilon' = \epsilon_z/2$, and call (ϵ', ξ, n) -FMM to evaluate all the functions on all the points simultaneously in

$$O\left(n \log(\frac{1}{\epsilon'}) \log^\xi(\frac{n}{\tau \epsilon'})\right) = O\left(n \log(\frac{1}{\epsilon_z}) \log^\xi(\frac{n}{\tau \epsilon_z})\right)$$

arithmetic operations, such that, for all d_i , $\tilde{\Phi}(d_i) = \Phi(d_i) + \epsilon'_i$, where $|\epsilon'_i| \leq \epsilon'$. Then

$$|\hat{\mathbf{z}}'_i| = \sqrt{(d_i - \hat{\lambda}_1)(\hat{\lambda}_n - d_i)} \exp(\tilde{\Phi}(d_i)) = \sqrt{(d_i - \hat{\lambda}_1)(\hat{\lambda}_n - d_i)} \exp(\Phi(d_i)) \exp(\epsilon'_i) = |\hat{\mathbf{z}}_i| \exp(\epsilon'_i).$$

Recall that for $1 + 2x \leq \exp(x)$ for $-1/4 \leq x \leq 0$ and $\exp(x) \leq 1 + 2x$ for $0 \leq x \leq 1/4$, which means that $\exp(\epsilon'_i) \in 1 \pm 2\epsilon'$, since $\epsilon' = \epsilon_z/2 \in (0, 1/4)$. This finally gives $\|\hat{\mathbf{z}} - \hat{\mathbf{z}}'\| \leq \epsilon_z \|\hat{\mathbf{z}}\|$.

The matrix $\hat{\mathbf{H}}' = \begin{pmatrix} \hat{\alpha} & \hat{\mathbf{z}}'^\top \\ \hat{\mathbf{z}}' & \mathbf{D} \end{pmatrix}$ satisfies

$$\|\mathbf{H} - \hat{\mathbf{H}}'\| \leq \|\mathbf{H} - \hat{\mathbf{H}}\| + \|\hat{\mathbf{H}} - \hat{\mathbf{H}}'\| \leq \frac{\epsilon \tau^3}{2} + \frac{n\epsilon}{1-n\epsilon} + \|\hat{\mathbf{z}} - \hat{\mathbf{z}}'\| \leq \frac{\epsilon \tau^3}{2} + \frac{n\epsilon}{1-n\epsilon} + \epsilon_z \|\hat{\mathbf{z}}\|.$$

From Lemma A.3 we have that $\|\hat{\mathbf{z}}\| \leq \|\mathbf{z}\| + \|\mathbf{z} - \hat{\mathbf{z}}\| \leq 1 + \frac{n\epsilon}{1-n\epsilon}$ which gives the final bounds. \square

B.4 Approximating inner products with the eigenvectors

In the last part of the analysis we provide bounds on the errors for inner products of the form $\hat{\mathbf{u}}_i^\top \mathbf{q}$, between the i -th eigenvector and some arbitrary vector \mathbf{q} . Following the procedure of Section 5 in [48], we write

$$\hat{\mathbf{u}}_i^\top \mathbf{q} = \frac{-\mathbf{q}_1 + \Phi(\hat{\lambda}_i)}{\sqrt{1 + \Psi(\hat{\lambda}_i)}},$$

where $\Phi(\lambda) = \sum_{k=2}^n \frac{\hat{\mathbf{z}}_k \mathbf{q}_k}{d_k - \lambda}$ and $\Psi(\lambda) = \sum_{k=2}^n \frac{\hat{\mathbf{z}}_k^2}{(d_k - \lambda)^2}$. To compute all the products for all i with \mathbf{q} , we can use FMM to approximate $\Phi(\lambda)$ and $\Psi(\lambda)$ on n points.

Lemma B.3. Let $\mathbf{H} = \begin{pmatrix} \alpha & \mathbf{z}^\top \\ \mathbf{z} & \mathbf{D} \end{pmatrix}$ be a symmetric arrowhead matrix with $\|\mathbf{H}\| \leq 1$, that satisfies the requirements of Lemma A.1 with parameter τ , and let $d_2 < d_3 < \dots < d_n$ be the diagonal elements of \mathbf{D} . Let $\epsilon_z, \widehat{\lambda}_i, \widehat{\mathbf{z}}, \widehat{\mathbf{z}}', \widehat{\mathbf{H}},$ and $\widehat{\mathbf{H}}'$ be the same as in Lemma B.2. Let \mathbf{q} be a fixed vector with $\|\mathbf{q}\| \leq 1$, and $\widehat{\mathbf{u}}_i, i \in [n]$, be the eigenvectors of $\widehat{\mathbf{H}}$. Assuming an (ϵ, n) -FMM, we can approximate all the inner products $\widehat{\mathbf{u}}_i^\top \mathbf{q}$, by some values x_i such that

$$|\widehat{\mathbf{u}}_i^\top \mathbf{q} - x_i| \leq 147\epsilon_z \frac{(n+1)^2}{\tau^6},$$

for all $i \in [n]$ simultaneously, in $O\left(n \log\left(\frac{1}{\epsilon_z}\right) \log^\xi\left(\frac{n}{\tau\epsilon_z}\right)\right)$ arithmetic operations.

Proof. First, let us rewrite the functions Φ and Ψ to account for the errors in $\widehat{\mathbf{z}}'$. Recall that from Lemma B.2 it holds that $|\widehat{\mathbf{z}}_i - \widehat{\mathbf{z}}'_i| \leq \epsilon_z |\widehat{\mathbf{z}}_i|$, in which case we can write

$$\Phi'(\lambda) = \sum_{k=2}^n \frac{(1 + \epsilon_k) \widehat{\mathbf{z}}_k \mathbf{q}_k}{d_k - \lambda}, \quad \text{and} \quad \Psi'(\lambda) = \sum_{k=2}^n \frac{(1 + \epsilon_k)^2 \widehat{\mathbf{z}}_k^2}{(d_k - \lambda)^2},$$

where $|\epsilon_k| \leq \epsilon_z$. Let $\widehat{\mathbf{u}}'_i$ be the i -th eigenvector of $\widehat{\mathbf{H}}'$. Now we can consider the following bound:

$$\begin{aligned} |\widehat{\mathbf{u}}_i^\top \mathbf{q} - \widehat{\mathbf{u}}'_i{}^\top \mathbf{q}| &= \left| \frac{-\mathbf{q}_1 + \Phi(\widehat{\lambda}_i)}{\sqrt{1 + \Psi(\widehat{\lambda}_i)}} - \frac{-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)}{\sqrt{1 + \Psi'(\widehat{\lambda}_i)}} \right| \\ &= \left| \frac{(-\mathbf{q}_1 + \Phi(\widehat{\lambda}_i))\sqrt{1 + \Psi'(\widehat{\lambda}_i)} - (-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i))\sqrt{1 + \Psi(\widehat{\lambda}_i)}}{\sqrt{1 + \Psi(\widehat{\lambda}_i)}\sqrt{1 + \Psi'(\widehat{\lambda}_i)}} \right| \\ &= \left| \frac{\Phi(\widehat{\lambda}_i)\sqrt{1 + \Psi'(\widehat{\lambda}_i)} - \Phi'(\widehat{\lambda}_i)\sqrt{1 + \Psi(\widehat{\lambda}_i)} + \mathbf{q}_1 \left(\sqrt{1 + \Psi(\widehat{\lambda}_i)} - \sqrt{1 + \Psi'(\widehat{\lambda}_i)} \right)}{\sqrt{1 + \Psi(\widehat{\lambda}_i)}\sqrt{1 + \Psi'(\widehat{\lambda}_i)}} \right| \\ &\leq \frac{\left| \Phi(\widehat{\lambda}_i)\sqrt{1 + \Psi'(\widehat{\lambda}_i)} - \Phi'(\widehat{\lambda}_i)\sqrt{1 + \Psi(\widehat{\lambda}_i)} \right| + \left| \mathbf{q}_1 \left(\sqrt{1 + \Psi(\widehat{\lambda}_i)} - \sqrt{1 + \Psi'(\widehat{\lambda}_i)} \right) \right|}{\left| \sqrt{1 + \Psi(\widehat{\lambda}_i)}\sqrt{1 + \Psi'(\widehat{\lambda}_i)} \right|} \end{aligned} \quad (20)$$

Let $\zeta_k = \frac{\widehat{\mathbf{z}}_k}{d_k - \widehat{\lambda}_i}$. Assuming $\epsilon_z \in (0, 1/2)$, we get the following lower bound for the denominator

$$\begin{aligned} \left| \sqrt{1 + \Psi(\widehat{\lambda}_i)}\sqrt{1 + \Psi'(\widehat{\lambda}_i)} \right| &= \left| \sqrt{1 + \sum_{k=2}^n \zeta_k^2} \sqrt{1 + \sum_{k=2}^n (1 + \epsilon_k)^2 \zeta_k^2} \right| \\ &\geq \left| \sqrt{1 + \sum_{k=2}^n \zeta_k^2} \sqrt{1 + \frac{1}{4} \sum_{k=2}^n \zeta_k^2} \right| \\ &\geq \frac{1}{2} \left| 1 + \sum_{k=2}^n \zeta_k^2 \right| \\ &= \frac{1}{2} \left| 1 + \Psi(\widehat{\lambda}_i) \right|. \end{aligned} \quad (21)$$

For the right part of the numerator, we have

$$\left| \mathbf{q}_1 \left(\sqrt{1 + \Psi(\widehat{\lambda}_i)} - \sqrt{1 + \Psi'(\widehat{\lambda}_i)} \right) \right| \leq \left| \sqrt{1 + \sum_{k=2}^n \zeta_k^2} - \sqrt{1 + \sum_{k=2}^n (1 + \epsilon_k)^2 \zeta_k^2} \right|$$

$$\begin{aligned}
&\leq \left| 1 + \sum_{k=2}^n \zeta_k^2 - 1 - \sum_{k=2}^n (1 + \epsilon_k)^2 \zeta_k^2 \right| \\
&\leq \left| \sum_{k=2}^n \epsilon_k (2 + \epsilon_k) \zeta_k^2 \right| \\
&\leq 3\epsilon_z \sum_{k=2}^n \zeta_k^2 \\
&= 3\epsilon_z \Psi(\widehat{\lambda}_i).
\end{aligned} \tag{22}$$

For the left part of the numerator, we have

$$\begin{aligned}
&\left| \Phi(\widehat{\lambda}_i) \sqrt{1 + \Psi'(\widehat{\lambda}_i)} - \Phi'(\widehat{\lambda}_i) \sqrt{1 + \Psi(\widehat{\lambda}_i)} \right| \\
&= \left| \left(\sum_{k=2}^n \zeta_k \mathbf{q}_k \right) \sqrt{1 + \sum_{k=2}^n (1 + \epsilon_k)^2 \zeta_k^2} - \left(\sum_{k=2}^n (1 + \epsilon_k) \zeta_k \mathbf{q}_k \right) \sqrt{1 + \sum_{k=2}^n \zeta_k^2} \right| \\
&= \left| \sum_{k=2}^n \left(\zeta_k \mathbf{q}_k \left[\sqrt{1 + \sum_{l=2}^n (1 + \epsilon_l)^2 \zeta_l^2} - (1 + \epsilon_k) \sqrt{1 + \sum_{l=2}^n \zeta_l^2} \right] \right) \right| \\
&= \left| \sum_{k=2}^n \left(\zeta_k \mathbf{q}_k \frac{1 + \sum_{l=2}^n (1 + \epsilon_l)^2 \zeta_l^2 - (1 + \epsilon_k)^2 \left(1 + \sum_{l=2}^n \zeta_l^2 \right)}{\sqrt{1 + \sum_{l=2}^n (1 + \epsilon_l)^2 \zeta_l^2} + (1 + \epsilon_k) \sqrt{1 + \sum_{l=2}^n \zeta_l^2}} \right) \right| \\
&= \left| \sum_{k=2}^n \left(\zeta_k \mathbf{q}_k \frac{1 - (1 + \epsilon_k)^2 + \sum_{l=2}^n (1 + \epsilon_l)^2 \zeta_l^2 - \sum_{l=2}^n (1 + \epsilon_k)^2 \zeta_l^2}{\sqrt{1 + \sum_{l=2}^n (1 + \epsilon_l)^2 \zeta_l^2} + (1 + \epsilon_k) \sqrt{1 + \sum_{l=2}^n \zeta_l^2}} \right) \right| \\
&= \left| \sum_{k=2}^n \left(\zeta_k \mathbf{q}_k \frac{1 - (1 + \epsilon_k)^2 + \sum_{l=2}^n \zeta_l^2 [(1 + \epsilon_l)^2 - (1 + \epsilon_k)^2]}{\sqrt{1 + \sum_{l=2}^n (1 + \epsilon_l)^2 \zeta_l^2} + (1 + \epsilon_k) \sqrt{1 + \sum_{l=2}^n \zeta_l^2}} \right) \right| \\
&= \left| \sum_{k=2}^n \left(\zeta_k \mathbf{q}_k \frac{-2\epsilon_k - \epsilon_k^2 + \sum_{l=2}^n \zeta_l^2 [2\epsilon_l + \epsilon_l^2 - 2\epsilon_k + \epsilon_k^2]}{\sqrt{1 + \sum_{l=2}^n (1 + \epsilon_l)^2 \zeta_l^2} + (1 + \epsilon_k) \sqrt{1 + \sum_{l=2}^n \zeta_l^2}} \right) \right| \\
&\leq \sum_{k=2}^n \left(|\zeta_k \mathbf{q}_k| \frac{|-2\epsilon_k - \epsilon_k^2 + \sum_{l=2}^n \zeta_l^2 [2\epsilon_l + \epsilon_l^2 - 2\epsilon_k + \epsilon_k^2]|}{\sqrt{1 + \sum_{l=2}^n (1 + \epsilon_l)^2 \zeta_l^2} + (1 + \epsilon_k) \sqrt{1 + \sum_{l=2}^n \zeta_l^2}} \right). \tag{23}
\end{aligned}$$

Once again, we lower bound the denominator and upper bound the numerator for Eq. (23). For the denominator we have:

$$\begin{aligned}
\left| \sqrt{1 + \sum_{l=2}^n (1 + \epsilon_l)^2 \zeta_l^2} + (1 + \epsilon_k) \sqrt{1 + \sum_{l=2}^n \zeta_l^2} \right| &\geq \left| \sqrt{1 + \frac{1}{4} \sum_{l=2}^n \zeta_l^2} + \frac{1}{2} \sqrt{1 + \sum_{l=2}^n \zeta_l^2} \right| \\
&\geq \sqrt{1 + \sum_{l=2}^n \zeta_l^2} \\
&= \sqrt{1 + \Psi(\widehat{\lambda}_i)}.
\end{aligned} \tag{24}$$

For the numerator it holds that:

$$\left| -2\epsilon_k - \epsilon_k^2 + \sum_{l=2}^n \zeta_l^2 [2\epsilon_l + \epsilon_l^2 - 2\epsilon_k + \epsilon_k^2] \right| \leq 3\epsilon_z + 6\epsilon_z \sum_{l=2}^n \zeta_l^2 = 3\epsilon_z + 6\epsilon_z \Psi(\widehat{\lambda}_i). \quad (25)$$

Finally, recall that from Lemma A.1 it holds that

$$\begin{aligned} |\zeta_k \mathbf{q}_k| &= \left| \frac{\mathbf{z}_k \mathbf{q}_k}{d_k - \widehat{\lambda}_i} \right| \leq \frac{2(n+1)}{\tau^3}, \\ \Psi(\widehat{\lambda}_i) &= \sum_{k=2}^n \zeta_k^2 = \sum_{k=2}^n \frac{\mathbf{z}_k^2}{(d_k - \widehat{\lambda}_i)^2} \leq \sum_{k=2}^n \frac{4(n+1)^2}{\tau^6} \end{aligned} \quad (26)$$

Combining (20), (21), (22), (23), (24), (25), and (26), we obtain

$$\begin{aligned} |\widehat{\mathbf{u}}_i^\top \mathbf{q} - \widehat{\mathbf{u}}_i'^\top \mathbf{q}| &\leq \frac{2}{|1 + \Psi(\widehat{\lambda}_i)|} \left[3\epsilon_z \Psi(\widehat{\lambda}_i) + \sum_{k=2}^n \left(|\zeta_k \mathbf{q}_k| \frac{3\epsilon_z + 6\epsilon_z \Psi(\widehat{\lambda}_i)}{\sqrt{1 + \Psi(\widehat{\lambda}_i)}} \right) \right] \\ &\leq 6\epsilon_z \Psi(\widehat{\lambda}_i) + 12\epsilon_z \sum_{k=2}^n |\zeta_k \mathbf{q}_k| \\ &\leq 6\epsilon_z \frac{4(n+1)^2}{\tau^6} + 12\epsilon_z (n-1) \frac{2(n+1)}{\tau^3} \\ &\leq 48\epsilon_z \frac{(n+1)^2}{\tau^6}. \end{aligned} \quad (27)$$

Next, we can choose some error parameter ϵ_{FMM} and include the errors introduced by using (ϵ_{FMM}, ξ, n) -FMM to approximate Φ' and Ψ' (which are also listed in Equations (15), (16)). The evaluation points satisfy the corresponding requirements. We shall denote by ϕ'_i the FMM error in $\Phi'(\widehat{\lambda}_i)$ and ψ'_i the FMM error in $\Psi'(\widehat{\lambda}_i)$, such that $|\psi_i|, |\phi_i| \leq \epsilon_{FMM}$. In this case we will write

$$\widehat{\mathbf{u}}_i''^\top \mathbf{q} = \frac{-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i) + \phi_i}{\sqrt{1 + \Psi'(\widehat{\lambda}_i) + \psi_i}} = \frac{-\mathbf{q}_1 + \sum_{k=2}^n \frac{(1+\epsilon_k)\widehat{\mathbf{z}}_k \mathbf{q}_k}{d_k - \widehat{\lambda}_i} + \phi_i}{\sqrt{1 + \sum_{k=2}^n \frac{(1+\epsilon_k)^2 \mathbf{z}_k^2}{d_k - \widehat{\lambda}_i} + \psi_i}}.$$

The error is bounded as follows.

$$\begin{aligned} |\widehat{\mathbf{u}}_i''^\top \mathbf{q} - \widehat{\mathbf{u}}_i'^\top \mathbf{q}| &= \left| \frac{-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)}{\sqrt{1 + \Psi'(\widehat{\lambda}_i)}} - \frac{-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i) + \phi_i}{\sqrt{1 + \Psi'(\widehat{\lambda}_i) + \psi_i}} \right| \\ &= \left| \frac{(-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)) \sqrt{1 + \Psi'(\widehat{\lambda}_i) + \psi_i} - (-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i) + \phi_i) \sqrt{1 + \Psi'(\widehat{\lambda}_i)}}{\sqrt{1 + \Psi'(\widehat{\lambda}_i)} \sqrt{1 + \Psi'(\widehat{\lambda}_i) + \psi_i}} \right| \\ &= \left| \frac{(-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)) \left(\sqrt{1 + \Psi'(\widehat{\lambda}_i) + \psi_i} - \sqrt{1 + \Psi'(\widehat{\lambda}_i)} \right) - \phi_i \sqrt{1 + \Psi'(\widehat{\lambda}_i)}}{\sqrt{1 + \Psi'(\widehat{\lambda}_i)} \sqrt{1 + \Psi'(\widehat{\lambda}_i) + \psi_i}} \right| \\ &\leq \left| \frac{(-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)) \left(\sqrt{1 + \Psi'(\widehat{\lambda}_i) + \psi_i} - \sqrt{1 + \Psi'(\widehat{\lambda}_i)} \right)}{\sqrt{1 + \Psi'(\widehat{\lambda}_i)} \sqrt{1 + \Psi'(\widehat{\lambda}_i) + \psi_i}} \right| + \left| \frac{\phi_i}{\sqrt{1 + \Psi'(\widehat{\lambda}_i) + \psi_i}} \right| \end{aligned}$$

$$\leq \left| \frac{\left(-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)\right) \left(\sqrt{1 + \Psi'(\widehat{\lambda}_i)} + \psi_i - \sqrt{1 + \Psi'(\widehat{\lambda}_i)}\right)}{\sqrt{1 + \Psi'(\widehat{\lambda}_i)} \sqrt{1 + \Psi'(\widehat{\lambda}_i)} + \psi_i} \right| + \epsilon_{FMM}. \quad (28)$$

For the demoninator of the leftmost term we have

$$\left| \sqrt{1 + \Psi'(\widehat{\lambda}_i)} \sqrt{1 + \Psi'(\widehat{\lambda}_i)} + \psi_i \right| \geq \frac{1}{\sqrt{2}} \left| 1 + \Psi'(\widehat{\lambda}_i) \right|. \quad (29)$$

For the numerator we have

$$\begin{aligned} & \left| \left(-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)\right) \left(\sqrt{1 + \Psi'(\widehat{\lambda}_i)} + \psi_i - \sqrt{1 + \Psi'(\widehat{\lambda}_i)}\right) \right| \\ &= \left| \left(-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)\right) \frac{\left(1 + \Psi'(\widehat{\lambda}_i) + \psi_i - 1 - \Psi'(\widehat{\lambda}_i)\right)}{\left(\sqrt{1 + \Psi'(\widehat{\lambda}_i)} + \psi_i + \sqrt{1 + \Psi'(\widehat{\lambda}_i)}\right)} \right| \\ &= \left| \left(-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)\right) \frac{\psi_i}{\left(\sqrt{1 + \Psi'(\widehat{\lambda}_i)} + \psi_i + \sqrt{1 + \Psi'(\widehat{\lambda}_i)}\right)} \right| \\ &\leq \left| \left(-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)\right) \frac{\psi_i}{\left(\frac{1}{\sqrt{2}} + 1\right) \sqrt{1 + \Psi'(\widehat{\lambda}_i)}} \right| \\ &\leq \epsilon_{FMM} \left| \frac{-\mathbf{q}_1 + \Phi'(\widehat{\lambda}_i)}{\sqrt{1 + \Psi'(\widehat{\lambda}_i)}} \right| \\ &= \epsilon_{FMM} \left| \widehat{\mathbf{u}}_i'^{\top} \mathbf{q} \right|. \end{aligned} \quad (30)$$

Combining the bounds of (28), (29), and (30), we obtain

$$\left| \widehat{\mathbf{u}}_i''^{\top} \mathbf{q} - \widehat{\mathbf{u}}_i'^{\top} \mathbf{q} \right| \leq \frac{\epsilon_{FMM} \left| \widehat{\mathbf{u}}_i'^{\top} \mathbf{q} \right|}{\frac{1}{\sqrt{2}} \left| 1 + \Psi'(\widehat{\lambda}_i) \right|} + \epsilon_{FMM} \leq \epsilon_{FMM} \left(1 + \sqrt{2} \left| \widehat{\mathbf{u}}_i'^{\top} \mathbf{q} \right| \right). \quad (31)$$

Finally, the bounds of (27) and (31) are combined to provide

$$\begin{aligned} \left| \widehat{\mathbf{u}}_i^{\top} \mathbf{q} - \widehat{\mathbf{u}}_i''^{\top} \mathbf{q} \right| &\leq \left| \widehat{\mathbf{u}}_i^{\top} \mathbf{q} - \widehat{\mathbf{u}}_i'^{\top} \mathbf{q} \right| + \left| \widehat{\mathbf{u}}_i'^{\top} \mathbf{q} - \widehat{\mathbf{u}}_i''^{\top} \mathbf{q} \right| \\ &\leq 48\epsilon_z \frac{(n+1)^2}{\tau^6} + \epsilon_{FMM} \left(1 + \sqrt{2} \left| \widehat{\mathbf{u}}_i'^{\top} \mathbf{q} \right| \right) \\ &\leq 48\epsilon_z \frac{(n+1)^2}{\tau^6} + \epsilon_{FMM} \left(1 + \sqrt{2} \left(1 + 48\epsilon_z \frac{(n+1)^2}{\tau^6} \right) \right) \\ &\leq 48\epsilon_z \frac{(n+1)^2}{\tau^6} + 3\epsilon_z + 96\epsilon_z^2 \frac{(n+1)^2}{\tau^6} \\ &= 147\epsilon_z \frac{(n+1)^2}{\tau^6}, \end{aligned}$$

where we set $\epsilon_{FMM} = \epsilon_z$. The total cost is

$$O \left(n \log \left(\frac{1}{\epsilon_{FMM}} \right) \log^{\xi} \left(\frac{n}{\tau \epsilon_{FMM}} \right) \right) = O \left(n \log \left(\frac{1}{\epsilon_z} \right) \log^{\xi} \left(\frac{n}{\tau \epsilon_z} \right) \right).$$

□

B.5 Proof of Theorem 2.1

We can finally combine all the results to prove Theorem 2.1, which we restate below for readability.

Theorem B.1. *Given a symmetric arrowhead matrix $\mathbf{H} \in \mathbb{R}^{n \times n}$ as in Eq. (3), with $\|\mathbf{H}\| \leq 1$, an accuracy parameter $\epsilon \in (0, 1)$, a matrix \mathbf{B} with r columns $\mathbf{B}_i, i \in [r]$, where $\|\mathbf{B}_i\| \leq 1$, and an (ϵ, n) -FMM implementation (see Prop. 1.1), we can compute a diagonal matrix $\tilde{\Lambda}$, and a matrix $\tilde{\mathbf{Q}}_{\mathbf{B}}$, such that*

$$\left\| \mathbf{H} - \mathbf{Q} \tilde{\Lambda} \mathbf{Q}^\top \right\| \leq \epsilon, \quad \left| \left(\mathbf{Q}^\top \mathbf{B} - \tilde{\mathbf{Q}}_{\mathbf{B}} \right)_{i,j} \right| \leq \epsilon/n^2,$$

where $\mathbf{Q} \in \mathbb{R}^{n \times n}$ is (exactly) orthogonal, in $O\left(nr \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$ arithmetic operations and comparisons.

Alternatively, if only want to compute a set of approximate values $\tilde{\lambda}_1, \dots, \tilde{\lambda}_n$, such that $|\lambda_i(\mathbf{H}) - \tilde{\lambda}_i| \leq \epsilon$, the complexity reduces to $O\left(n \log\left(\frac{1}{\epsilon}\right) \log^{\xi}\left(\frac{n}{\epsilon}\right)\right)$ arithmetic operations.

Proof. Step 1: Deflation. The first step is to use deflation in order to ensure that the requirements of Lemma A.1 are satisfied. To that end we apply Proposition A.1 from the Appendix with parameter τ . This way we obtain a symmetric matrix $\tilde{\mathbf{H}} = \begin{pmatrix} \tilde{\mathbf{H}}' & 0 \\ 0 & \tilde{\mathbf{D}} \end{pmatrix}$, such that $\tilde{\mathbf{D}}$ is diagonal, and the upper-left block $\tilde{\mathbf{H}}' = \begin{pmatrix} \tilde{\alpha}' & \tilde{\mathbf{z}}'^\top \\ \tilde{\mathbf{z}}' & \tilde{\mathbf{D}}' \end{pmatrix}$ is a symmetric arrowhead matrix that satisfies the requirements of Lemma A.1, namely,

$$\tilde{\mathbf{D}}'_{j+1,j+1} - \tilde{\mathbf{D}}'_{j,j} \geq \tau, \quad \text{and} \quad |\tilde{\mathbf{z}}'_i| \geq \tau. \quad (32)$$

The matrix $\tilde{\mathbf{H}}$ satisfies $\|\mathbf{H} - \mathbf{G} \tilde{\mathbf{H}} \mathbf{G}^\top\| \leq n\tau$, where \mathbf{G} is orthogonal. Since \mathbf{G} is orthogonal, $\tilde{\mathbf{H}}$ is similar to $\mathbf{G} \tilde{\mathbf{H}} \mathbf{G}^\top$, i.e., they have the same eigenvalues. Then Weyl's inequality gives

$$|\lambda_i(\mathbf{H}) - \lambda_i(\tilde{\mathbf{H}})| = |\lambda_i(\mathbf{H}) - \lambda_i(\mathbf{G} \tilde{\mathbf{H}} \mathbf{G}^\top)| \leq \|\mathbf{H} - \mathbf{G} \tilde{\mathbf{H}} \mathbf{G}^\top\| \leq n\tau.$$

As described in Proposition A.1, $\tilde{\mathbf{H}}'$ is a principal submatrix of \mathbf{H} , and therefore due to the eigenvalue interlacing property it holds that $\|\tilde{\mathbf{H}}'\| \leq \|\mathbf{H}\| \leq 1$.

Step 2: Diagonalization of $\tilde{\mathbf{H}}$. In the second step we focus on $\tilde{\mathbf{H}}$. There are two different cases, depending on whether we are interested only on the eigenvalues, or if we also want to multiply the eigenvector matrix with another matrix.

Case 1 - Eigenvalues only. If we only care about the eigenvalues, we can ignore the computation of \mathbf{G} , and therefore the deflation costs $O(n \log(n))$ operations. It remains to approximate the eigenvalues of $\tilde{\mathbf{H}}'$. For this we use Lemma B.1 with error parameter ϵ_λ , which returns some values $\tilde{\lambda}_1, \dots, \tilde{\lambda}_n$ in $O\left(n \log\left(\frac{1}{\epsilon_\lambda}\right) \log^{\xi}\left(\frac{n}{\epsilon_\lambda}\right)\right)$ arithmetic operations such that $|\tilde{\lambda}_i - \lambda_i(\tilde{\mathbf{H}}')| \leq \epsilon_\lambda$. If we set:

$$\tau = \frac{\epsilon}{2n}, \quad \text{and} \quad \epsilon_\lambda = \frac{\epsilon}{2},$$

a triangle inequality gives that $|\tilde{\lambda}_i - \lambda_i(\mathbf{H})| \leq n\tau + \epsilon_\lambda \leq \epsilon$. The complexity of Lemma B.1 becomes $O\left(n \log\left(\frac{1}{\epsilon}\right) \log^{\xi}\left(\frac{n}{\epsilon}\right)\right)$, which gives a matching total number of arithmetic operations and comparisons.

Case 2 - Eigenvalues and eigenvectors. In this case we do not only need the eigenvalues, but we are also given a matrix \mathbf{B} that needs to be multiplied from the left with the eigenvector matrix. We assume that the matrix product $\mathbf{G}_{\mathbf{B}} = \mathbf{G}^\top \mathbf{B}$ is already formed on-the-fly in a total of $O(nr)$ operations from Proposition A.1 in Step 1.

We then proceed with the diagonalization of $\tilde{\mathbf{H}}'$. The following steps are used to diagonalize $\tilde{\mathbf{H}}'$:

1. We first set $\epsilon_\lambda = \epsilon' \frac{\tau^3}{2(n+1)}$, for some $\epsilon' \in (0, 1/n)$ that is specified later, and use Lemma B.1, to approximate all the eigenvalues of $\tilde{\mathbf{H}}'$ in $O\left(n \log\left(\frac{1}{\epsilon_\lambda}\right) \log^{\xi}\left(\frac{n}{\epsilon_\lambda}\right)\right) = O\left(n \log^{\xi+1}\left(\frac{n}{\epsilon'}\right)\right)$ arithmetic operations. The returned eigenvalues $\hat{\lambda}_i$ satisfy $|\hat{\lambda}_i - \lambda_i(\tilde{\mathbf{H}}')| \leq \epsilon_\lambda$. Note that this value of ϵ_λ satisfies the requirement of Lemmas A.3 and B.2, and therefore we can use them.

From Lemma A.3, we know that $\hat{\lambda}_i$ are the exact eigenvalues of the matrix $\hat{\mathbf{H}} = \begin{pmatrix} \hat{\alpha} & \hat{\mathbf{z}}^\top \\ \hat{\mathbf{z}} & \hat{\mathbf{D}} \end{pmatrix}$, where:

$$|\tilde{\alpha}' - \hat{\alpha}| \leq \frac{\epsilon' \tau^3}{2}, \quad \|\tilde{\mathbf{z}}' - \hat{\mathbf{z}}\| \leq \frac{n\epsilon'}{1 - n\epsilon'}, \quad \|\tilde{\mathbf{H}}' - \hat{\mathbf{H}}\| \leq \frac{\epsilon' \tau^3}{2} + \frac{n\epsilon'}{1 - n\epsilon'}.$$

2. We now compute the elements of $\hat{\mathbf{H}}$. The value of $\hat{\alpha}$ can be computed exactly in $O(n)$ operations from Lemma A.2. The vector $\hat{\mathbf{z}}$ can be approximated using Lemma B.2 in $O\left(n \log(\frac{1}{\epsilon_z}) \log^\xi(\frac{n}{\tau\epsilon_z})\right)$ arithmetic operations, where $\epsilon_z \in (0, 1/2)$ is some chosen input parameter (specified below). The returned vector $\tilde{\mathbf{z}}'$ satisfies

$$\|\tilde{\mathbf{z}}' - \hat{\mathbf{z}}\| \leq \epsilon_z \|\hat{\mathbf{z}}\|.$$

3. Next, we partition the matrix $\mathbf{G}_\mathbf{B} = \mathbf{G}^\top \mathbf{B}$, which was computed in the previous steps, in the form $\mathbf{G}^\top \mathbf{B} = \begin{pmatrix} \mathbf{G}_1^\top \mathbf{B} \\ \mathbf{G}_2^\top \mathbf{B} \end{pmatrix}$.

We then apply Lemma B.3 on $(\mathbf{G}_1^\top \mathbf{B})$. Specifically, given the approximate eigenvalues $\hat{\lambda}_i$ and the approximate vector $\tilde{\mathbf{z}}'$, we use Lemma B.3 to approximate the matrix product $\hat{\mathbf{U}}^\top (\mathbf{G}_1^\top \mathbf{B})$ in $O\left(nr \log(\frac{1}{\epsilon_z}) \log^\xi(\frac{n}{\tau\epsilon_z})\right)$ operations, where $\hat{\mathbf{U}}$ denotes the orthogonal matrix whose columns are the eigenvectors of $\hat{\mathbf{H}}$ ($\hat{\mathbf{U}}$ is not computed explicitly). Let $\mathbf{X}_\mathbf{B} = \hat{\mathbf{U}}^\top \mathbf{G}_1^\top \mathbf{B}$ be the true matrix product, and $\mathbf{X}'_\mathbf{B}$ be the matrix returned by Lemma B.3, respectively. Then for all i, j , $|(X_\mathbf{B})_{i,j} - (X'_\mathbf{B})_{i,j}| \leq 147\epsilon_z \frac{(n+1)^2}{\tau^6}$.

4. We finally set $\tilde{\mathbf{Q}}_\mathbf{B} = \begin{pmatrix} \mathbf{X}'_\mathbf{B} \\ \mathbf{G}_2^\top \mathbf{B} \end{pmatrix}$, and $\tilde{\Lambda} = \begin{pmatrix} \hat{\Lambda} & \\ & \tilde{\mathbf{D}} \end{pmatrix}$.

Denote $\mathbf{Q} := \mathbf{G} \begin{pmatrix} \hat{\mathbf{U}} & \\ & \mathbf{I} \end{pmatrix}$. For the error $\|\mathbf{Q}^\top \mathbf{B} - \tilde{\mathbf{Q}}_\mathbf{B}\|$ we work as follows. Note that $\mathbf{Q}^\top \mathbf{B} = \begin{pmatrix} \hat{\mathbf{U}}^\top & \\ & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{G}_1^\top \mathbf{B} \\ \mathbf{G}_2^\top \mathbf{B} \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{U}}^\top \mathbf{G}_1^\top \mathbf{B} \\ \mathbf{G}_2^\top \mathbf{B} \end{pmatrix}$. Then for all i, j :

$$\left| (\mathbf{Q}^\top \mathbf{B} - \tilde{\mathbf{Q}}_\mathbf{B})_{i,j} \right| = \left| \begin{pmatrix} \hat{\mathbf{U}}^\top \mathbf{G}_1^\top \mathbf{B} \\ \mathbf{G}_2^\top \mathbf{B} \end{pmatrix}_{i,j} - \begin{pmatrix} \mathbf{X}'_\mathbf{B} \\ \mathbf{G}_2^\top \mathbf{B} \end{pmatrix}_{i,j} \right| = \left| \begin{pmatrix} \hat{\mathbf{U}}^\top \mathbf{G}_1^\top \mathbf{B} - \mathbf{X}'_\mathbf{B} \\ 0 \end{pmatrix}_{i,j} \right| = \left| \begin{pmatrix} \mathbf{X}_\mathbf{B} - \mathbf{X}'_\mathbf{B} \\ 0 \end{pmatrix}_{i,j} \right|.$$

The bottom part is zero. The top part was already bounded above as $|(X_\mathbf{B})_{i,j} - (X'_\mathbf{B})_{i,j}| \leq 147\epsilon_z \frac{(n+1)^2}{\tau^6}$.

We can now analyze the backward-error $\|\mathbf{H} - \mathbf{Q} \tilde{\Lambda} \mathbf{Q}^\top\|$. We have

$$\begin{aligned} \|\mathbf{H} - \mathbf{Q} \tilde{\Lambda} \mathbf{Q}^\top\| &= \left\| \mathbf{H} - \mathbf{G} \begin{pmatrix} \hat{\mathbf{U}} & \\ & \mathbf{I} \end{pmatrix} \tilde{\Lambda} \begin{pmatrix} \hat{\mathbf{U}} & \\ & \mathbf{I} \end{pmatrix}^\top \mathbf{G}^\top \right\| = \left\| \mathbf{H} - \mathbf{G} \begin{pmatrix} \hat{\mathbf{H}} & \\ & \tilde{\mathbf{D}} \end{pmatrix} \mathbf{G}^\top \right\| \\ &= \left\| \mathbf{H} - \mathbf{G} \begin{pmatrix} \tilde{\mathbf{H}}' + \hat{\mathbf{H}} - \tilde{\mathbf{H}}' & \\ & \tilde{\mathbf{D}} \end{pmatrix} \mathbf{G}^\top \right\| = \left\| \mathbf{H} - \mathbf{G} \tilde{\mathbf{H}} \mathbf{G}^\top + \mathbf{G}_1 (\hat{\mathbf{H}} - \tilde{\mathbf{H}}') \mathbf{G}_1^\top \right\| \\ &= \left\| \mathbf{H} - \mathbf{G} \tilde{\mathbf{H}} \mathbf{G}^\top \right\| + \left\| \hat{\mathbf{H}} - \tilde{\mathbf{H}}' \right\| \leq n\tau + \frac{\epsilon' \tau^3}{2} + \frac{n\epsilon'}{1 - n\epsilon'}. \end{aligned}$$

The next task is to set the parameters $\tau, \epsilon_\lambda, \epsilon'$, and ϵ_z and to upper bound the complexity. Given $\epsilon \in (0, 1/2)$ the desired error parameter, the rest are set as follows:

$$\tau = \frac{\epsilon}{2n}, \quad \epsilon' = \frac{\epsilon}{4n}, \quad \epsilon_\lambda = \epsilon' \frac{\tau^3}{2(n+1)} = \frac{\epsilon^4}{64n^4(n+1)}, \quad \epsilon_z = \epsilon \frac{\tau^6}{147n^2(n+1)^2} = \frac{\epsilon^7}{147 \cdot 64 \cdot n^8(n+1)^2}.$$

Then the errors become:

$$\begin{aligned}\|\mathbf{H} - \mathbf{Q}\tilde{\mathbf{A}}\mathbf{Q}^\top\| &\leq n\tau + \frac{\epsilon'\tau^3}{2} + \frac{n\epsilon'}{1-n\epsilon'} \leq \frac{\epsilon}{2} + \frac{\epsilon^4}{64n^4} + \frac{\epsilon/4}{1-\epsilon/4} \leq \epsilon, \\ \left|(\mathbf{Q}^\top\mathbf{B} - \tilde{\mathbf{Q}}_B)_{i,j}\right| &= \left|\begin{pmatrix} \mathbf{X}_B - \mathbf{X}'_B \\ 0 \end{pmatrix}_{i,j}\right| \leq 147\epsilon_z \frac{(n+1)^2}{\tau^6} = \epsilon/n^2,\end{aligned}$$

which also implies

$$\|\mathbf{Q}^\top\mathbf{B} - \tilde{\mathbf{Q}}^\top\| \leq \epsilon/n. \quad (33)$$

The complexity is as follows:

Deflation: $O(n \log(n) + nr)$,

Eigenvalues: $O\left(n \log\left(\frac{1}{\epsilon_\lambda}\right) \log^\xi\left(\frac{n}{\tau\epsilon_\lambda}\right)\right) = O\left(n \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$,

$\tilde{\mathbf{Z}}$: $O\left(n \log\left(\frac{1}{\epsilon_z}\right) \log^\xi\left(\frac{n}{\tau\epsilon_z}\right)\right) = O\left(n \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$,

$\tilde{\mathbf{Q}}_B$: $O\left(nr \log\left(\frac{1}{\epsilon_z}\right) \log^\xi\left(\frac{n}{\tau\epsilon_z}\right)\right) = O\left(nr \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$,

Total: $O\left(nr \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$.

□

C Tridiagonal diagonalization

C.1 Omitted proofs

The next lemma bounds the error of the reduction to arrowhead form when the spectral factorizations of the matrices \mathbf{T}_1 and \mathbf{T}_2 in Equation (2) are approximate rather than exact.

Lemma C.1 (Restatement of Lemma 2.1). *Let $\epsilon \in (0, 1/2)$ be a given accuracy parameter and $\mathbf{T} = \begin{pmatrix} \mathbf{T}_1 & \beta_{k+1}\mathbf{e}_k & \\ \beta_{k+1}\mathbf{e}_k^\top & \alpha_{k+1} & \beta_{k+2}\mathbf{e}_1^\top \\ & \beta_{k+2}\mathbf{e}_1 & \mathbf{T}_2 \end{pmatrix}$*

be a tridiagonal matrix with size $n \geq 3$ and $\|\mathbf{T}\| \leq 1$, where $\mathbf{T}_1 = \mathbf{U}_1\mathbf{D}_1\mathbf{U}_1^\top$ and $\mathbf{T}_2 = \mathbf{U}_2\mathbf{D}_2\mathbf{U}_2^\top$ be the exact spectral factorizations of \mathbf{T}_1 and \mathbf{T}_2 . Let $\tilde{\mathbf{U}}_1, \tilde{\mathbf{D}}_1, \tilde{\mathbf{U}}_2, \tilde{\mathbf{D}}_2$ be approximate spectral factorizations of $\mathbf{T}_1, \mathbf{T}_2$. If these factors satisfy

$$\left\|\mathbf{T}_{\{1,2\}} - \tilde{\mathbf{U}}_{\{1,2\}}\tilde{\mathbf{D}}_{\{1,2\}}\tilde{\mathbf{U}}_{\{1,2\}}^\top\right\| \leq \epsilon_1, \quad \left\|\tilde{\mathbf{U}}_{\{1,2\}}\tilde{\mathbf{U}}_{\{1,2\}}^\top - \mathbf{I}\right\| \leq \epsilon_1/n,$$

for some $\epsilon_1 \in (0, 1/2)$, where $\tilde{\mathbf{D}}_{\{1,2\}}$ are both diagonal, then, assuming an (ϵ, n) -FMM implementation as in Prop. 1.1, we can compute a diagonal matrix $\tilde{\mathbf{D}}$ and an approximately orthogonal matrix $\tilde{\mathbf{U}}$ such that

$$\left\|\tilde{\mathbf{U}}^\top\tilde{\mathbf{U}} - \mathbf{I}\right\| \leq 3(\epsilon_1 + \epsilon)/n, \quad \text{and} \quad \left\|\mathbf{T} - \tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{U}}^\top\right\| \leq 2\epsilon_1 + 7\epsilon,$$

in a total of $O\left(n^2 \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$ arithmetic operations and comparisons.

Proof. We first consider the matrix

$$\tilde{\mathbf{T}} = \begin{pmatrix} \tilde{\mathbf{U}}_1 & & \\ 1 & & \\ & \tilde{\mathbf{U}}_2 & \end{pmatrix} \begin{pmatrix} \alpha_{k+1} & \beta_{k+1}\tilde{\mathbf{l}}_1^\top & \beta_{k+2}\tilde{\mathbf{f}}_2^\top \\ \beta_{k+1}\tilde{\mathbf{l}}_1 & \tilde{\mathbf{D}}_1 & \\ \beta_{k+2}\tilde{\mathbf{f}}_2 & & \tilde{\mathbf{D}}_2 \end{pmatrix} \begin{pmatrix} & 1 & \\ \tilde{\mathbf{U}}_1^\top & & \\ & & \tilde{\mathbf{U}}_2^\top \end{pmatrix} = \mathbf{W}\tilde{\mathbf{H}}\mathbf{W}^\top,$$

where $\tilde{\mathbf{l}}_1^\top$ is the last row of $\tilde{\mathbf{U}}_1$ and $\tilde{\mathbf{f}}_2^\top$ is the first row of $\tilde{\mathbf{U}}_2$. $\tilde{\mathbf{H}}$ can be assembled in $O(n)$ operations, and \mathbf{W} in $O(n^2)$ operations. Note that $\mathbf{W}\tilde{\mathbf{H}}\mathbf{W}^\top$ is not a similarity transformation, since \mathbf{W} is not orthogonal, but it can be seen as an approximate similarity transformation, which is sufficient for our analysis.

In the next step, we need to use Theorem 2.1 to diagonalize $\tilde{\mathbf{H}}$, and multiply its eigenvector matrix with \mathbf{W} . Since neither $\tilde{\mathbf{H}}$ nor \mathbf{W} are guaranteed to have $\|\tilde{\mathbf{H}}\|, \|\mathbf{W}\| \leq 1$, we need to scale the appropriately so that we can apply Theorem 2.1. First note that the set of singular values of \mathbf{W} , $\Sigma(\mathbf{W})$, is equal to

$$\Sigma(\mathbf{W}) = \Sigma(\tilde{\mathbf{U}}_1) \cup \Sigma(\tilde{\mathbf{U}}_2) \cup \{1\}.$$

Since the singular values of $\tilde{\mathbf{U}}_1$ and $\tilde{\mathbf{U}}_2$ lie inside $[\sqrt{1 - \epsilon_1/n}, \sqrt{1 + \epsilon_1/n}]$ by assumption, then we have that $\sigma_i(\mathbf{W}) \in [\sqrt{1 - \epsilon_1/n}, \sqrt{1 + \epsilon_1/n}]$. This also implies that \mathbf{W} is invertible and that $\|\mathbf{W}\| \leq \sqrt{1 + \epsilon_1/n} < 2$. Next, consider the bound

$$\begin{aligned} \|\mathbf{T} - \tilde{\mathbf{T}}\| &= \left\| \begin{pmatrix} \mathbf{T}_1 & \beta_{k+1}\mathbf{e}_k & \\ \beta_{k+1}\mathbf{e}_k^\top & \alpha_{k+1} & \beta_{k+2}\mathbf{e}_1^\top \\ & \beta_{k+2}\mathbf{e}_1 & \mathbf{T}_2 \end{pmatrix} - \begin{pmatrix} \tilde{\mathbf{U}}_1\tilde{\mathbf{D}}_1\tilde{\mathbf{U}}_1^\top & \beta_{k+1}\tilde{\mathbf{U}}_1\tilde{\mathbf{I}}_1 & \\ \beta_{k+1}\tilde{\mathbf{I}}_1^\top\tilde{\mathbf{U}}_1^\top & \alpha_{k+1} & \beta_{k+2}\tilde{\mathbf{f}}_2^\top\tilde{\mathbf{U}}_2^\top \\ & \beta_{k+2}\tilde{\mathbf{U}}_2\tilde{\mathbf{f}}_2 & \tilde{\mathbf{U}}_2\tilde{\mathbf{D}}_2\tilde{\mathbf{U}}_2^\top \end{pmatrix} \right\| \\ &\leq \left\| \begin{pmatrix} \mathbf{T}_1 - \tilde{\mathbf{U}}_1\tilde{\mathbf{D}}_1\tilde{\mathbf{U}}_1^\top & & \\ & \mathbf{T}_2 - \tilde{\mathbf{U}}_2\tilde{\mathbf{D}}_2\tilde{\mathbf{U}}_2^\top & \end{pmatrix} \right\| + \|\mathbf{T}\| \left\| \begin{pmatrix} \tilde{\mathbf{I}}_1^\top\tilde{\mathbf{U}}_1^\top - \mathbf{e}_k^\top & \tilde{\mathbf{U}}_1\tilde{\mathbf{I}}_1 - \mathbf{e}_k & \\ & \tilde{\mathbf{f}}_2^\top\tilde{\mathbf{U}}_2^\top - \mathbf{e}_1^\top & \\ & \tilde{\mathbf{U}}_2\tilde{\mathbf{f}}_2 - \mathbf{e}_1 & \end{pmatrix} \right\| \\ &\leq \epsilon_1 + 2\epsilon_1/n. \end{aligned}$$

Then

$$\begin{aligned} \|\tilde{\mathbf{H}}\| &= \|\mathbf{W}^{-1}\mathbf{W}\tilde{\mathbf{H}}\mathbf{W}^\top\mathbf{W}^{-\top}\| \\ &\leq \|\mathbf{W}^{-1}\|^2 \|\mathbf{W}\tilde{\mathbf{H}}\mathbf{W}^\top\| \\ &= \|\mathbf{W}^{-1}\|^2 \|\tilde{\mathbf{T}}\| \\ &\leq \frac{1}{1 - \epsilon_1/n} (1 + \epsilon_1 + 2\epsilon_1/n) \\ &< 4. \end{aligned}$$

We can now apply Theorem 2.1 for the matrices $\tilde{\mathbf{H}}/4$ and $\mathbf{B} = \mathbf{W}/4$, which satisfy the norm requirements, and for error $\epsilon/4$. This way, in $O\left(n^2 \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$ operations, we can compute $\tilde{\mathbf{U}}, \tilde{\mathbf{\Lambda}}$, such that

$$\|\tilde{\mathbf{H}}/4 - \mathbf{Q}\tilde{\mathbf{\Lambda}}\mathbf{Q}^\top\| \leq \epsilon/4, \quad \|\mathbf{Q}^\top\mathbf{W}^\top/4 - \tilde{\mathbf{U}}^\top\| \leq \epsilon/4n,$$

where \mathbf{Q} is an (unknown) orthogonal matrix, and the rightmost bound comes from Eq. (33). If we rescale

$$\tilde{\mathbf{\Lambda}} \leftarrow 4\tilde{\mathbf{\Lambda}}, \quad \tilde{\mathbf{U}} \leftarrow 4\tilde{\mathbf{U}},$$

this gives

$$\|\tilde{\mathbf{H}} - \mathbf{Q}\tilde{\mathbf{\Lambda}}\mathbf{Q}^\top\| \leq \epsilon, \quad \|\mathbf{Q}^\top\mathbf{W}^\top - \tilde{\mathbf{U}}^\top\| \leq \epsilon/n.$$

Then we consider the error

$$\|\mathbf{T} - \tilde{\mathbf{U}}\tilde{\mathbf{\Lambda}}\tilde{\mathbf{U}}^\top\| \leq \|\mathbf{T} - \tilde{\mathbf{T}}\| + \|\tilde{\mathbf{T}} - \tilde{\mathbf{U}}\tilde{\mathbf{\Lambda}}\tilde{\mathbf{U}}^\top\|.$$

We can then bound the norm of $\tilde{\mathbf{\Lambda}}$ as follows:

$$\begin{aligned} \|\tilde{\mathbf{\Lambda}}\| &\leq \|\mathbf{Q}\tilde{\mathbf{\Lambda}}\mathbf{Q}^\top - \tilde{\mathbf{H}}\| + \|\tilde{\mathbf{H}}\| \\ &\leq \epsilon + \|\mathbf{W}^{-1}\|^2 \|\tilde{\mathbf{T}}\| \end{aligned}$$

$$\begin{aligned}
&\leq \epsilon + \left(\frac{1}{1-\epsilon_1/n}\right) \left(\|\mathbf{T}\| + \|\tilde{\mathbf{T}} - \mathbf{T}\|\right) \\
&\leq \epsilon + \left(\frac{1}{1-\epsilon_1/n}\right) (1 + \epsilon_1 + 2\epsilon_1/n) \\
&< 4.
\end{aligned}$$

Denoting $\mathbf{E}_U := \tilde{\mathbf{U}} - \mathbf{W}\mathbf{Q}$, we have that

$$\begin{aligned}
\|\tilde{\mathbf{T}} - \tilde{\mathbf{U}}\tilde{\mathbf{A}}\tilde{\mathbf{U}}^\top\| &= \|\mathbf{W}\tilde{\mathbf{H}}\mathbf{W}^\top - (\mathbf{W}\mathbf{Q} + \mathbf{E}_U)\tilde{\mathbf{A}}(\mathbf{W}\mathbf{Q} + \mathbf{E}_U)^\top\| \\
&\leq \|\mathbf{W}\tilde{\mathbf{H}}\mathbf{W}^\top - \mathbf{W}\mathbf{Q}\tilde{\mathbf{A}}\mathbf{Q}^\top\mathbf{W}^\top\| + 2\|\mathbf{E}_U\tilde{\mathbf{A}}\mathbf{Q}^\top\mathbf{W}^\top\| + \|\mathbf{E}_U\tilde{\mathbf{A}}\mathbf{E}_U^\top\| \\
&\leq \|\mathbf{W}\|^2 \|\tilde{\mathbf{H}} - \mathbf{Q}\tilde{\mathbf{A}}\mathbf{Q}^\top\| + 2\|\mathbf{E}_U\| \|\mathbf{W}\| \|\tilde{\mathbf{A}}\| + \|\mathbf{E}_U\|^2 \|\tilde{\mathbf{A}}\| \\
&\leq (1 + \epsilon_1/n)\epsilon + 2 \cdot 4(\epsilon/n)\sqrt{1 + \epsilon_1/n} + 4(\epsilon/n)^2 \\
&\leq 7\epsilon.
\end{aligned}$$

This finally gives

$$\|\mathbf{T} - \tilde{\mathbf{U}}\tilde{\mathbf{A}}\tilde{\mathbf{U}}^\top\| \leq \epsilon_1 + 2\epsilon_1/n + 7\epsilon \leq 2\epsilon_1 + 7\epsilon.$$

We also need to prove that $\tilde{\mathbf{U}}$ is approximately orthogonal. From the stability of singular values (which is a consequence of Weyl's inequality), we know that $|\sigma_i(\tilde{\mathbf{U}}) - \sigma_i(\mathbf{W}\mathbf{Q})| \leq \|\tilde{\mathbf{U}} - \mathbf{W}\mathbf{Q}\| \leq \epsilon/n$. The singular values of $\mathbf{W}\mathbf{Q}$ are the same as the singular values of \mathbf{W} since \mathbf{Q} is orthogonal. We already argued that $\sigma_i(\mathbf{W}) \in [\sqrt{1 - \epsilon_1/n}, \sqrt{1 + \epsilon_1/n}]$, and thus $\sigma_i(\tilde{\mathbf{U}}) \in [\sqrt{1 - \epsilon_1/n} - \epsilon/n, \sqrt{1 + \epsilon_1/n} + \epsilon/n] \in [1 - (\epsilon_1 + \epsilon)/n, 1 + (\epsilon_1 + \epsilon)/n]$, which gives roughly $\|\tilde{\mathbf{U}}\tilde{\mathbf{U}}^\top - \mathbf{I}\| \leq 3(\epsilon_1 + \epsilon)/n$. \square

Lemma C.2. Let $\epsilon \in (0, 1/2)$ be a given accuracy parameter and $\mathbf{T} = \begin{pmatrix} \mathbf{T}_1 & \beta_{k+1}\mathbf{e}_k & \\ \beta_{k+1}\mathbf{e}_k^\top & \alpha_{k+1} & \beta_{k+2}\mathbf{e}_1^\top \\ & \beta_{k+2}\mathbf{e}_1 & \mathbf{T}_2 \end{pmatrix}$ be a tridiagonal matrix with size $n \geq 2$, where $\mathbf{T}_1 = \mathbf{U}_1\mathbf{D}_1\mathbf{U}_1^\top$ and $\mathbf{T}_2 = \mathbf{U}_2\mathbf{D}_2\mathbf{U}_2^\top$ be the exact spectral factorizations of \mathbf{T}_1 and \mathbf{T}_2 . Let $\tilde{\mathbf{U}}_1, \tilde{\mathbf{D}}_1, \tilde{\mathbf{U}}_2, \tilde{\mathbf{D}}_2$ be approximate spectral factorizations of $\mathbf{T}_1, \mathbf{T}_2$. Assume that these factors satisfy

$$\|\mathbf{T}_{\{1,2\}} - \tilde{\mathbf{U}}_{\{1,2\}}\tilde{\mathbf{D}}_{\{1,2\}}\tilde{\mathbf{U}}_{\{1,2\}}^\top\| \leq \epsilon_1, \quad \|\tilde{\mathbf{U}}_{\{1,2\}}\tilde{\mathbf{U}}_{\{1,2\}}^\top - \mathbf{I}\| \leq \epsilon_1/n,$$

for some $\epsilon_1 \in (0, 1/2)$, where $\tilde{\mathbf{D}}_{\{1,2\}}$ are both diagonal. Assume also that $\tilde{\mathbf{D}}_1, \tilde{\mathbf{D}}_2$, as well as the last row $\tilde{\mathbf{f}}_1^\top$ of $\tilde{\mathbf{U}}_1$, and the first row $\tilde{\mathbf{f}}_2^\top$ of $\tilde{\mathbf{U}}_2$, are explicitly available.

Then we can compute a diagonal matrix $\tilde{\mathbf{D}}$, as well as the first row $\tilde{\mathbf{l}}$ and/or the last row $\tilde{\mathbf{f}}$ of an approximately orthogonal matrix $\tilde{\mathbf{U}}$, which satisfy

$$\|\tilde{\mathbf{U}}^\top\tilde{\mathbf{U}} - \mathbf{I}\| \leq 3(\epsilon_1 + \epsilon)/n, \quad \text{and} \quad \|\mathbf{T} - \tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{U}}^\top\| \leq 2\epsilon_1 + 7\epsilon,$$

in a total of $O\left(n \log^{\xi+1}\left(\frac{n}{\epsilon}\right)\right)$ arithmetic operations.

Proof. The main observation is that, if we only care about $\tilde{\mathbf{D}}$ and the first and/or last row of $\tilde{\mathbf{U}}$, then the proof of Lemma 2.1 can be simplified as follows. Since we have explicit access to $\tilde{\mathbf{l}}_1$ and $\tilde{\mathbf{f}}_2$, we can construct $\tilde{\mathbf{H}}$ in $O(n)$ arithmetic operations.

The next step is to apply Theorem 2.1 on $\tilde{\mathbf{H}}$. The theorem returns $\tilde{\mathbf{A}}$, which is a diagonal matrix with approximate eigenvalues, and $\tilde{\mathbf{U}}^\top$, which approximates the matrix product $\mathbf{Q}^\top\mathbf{W}^\top$. However, we do not need to compute the matrix product $\mathbf{Q}^\top\mathbf{W}^\top$. Instead, we only need to compute two matrix-vector products, $\mathbf{Q}^\top\mathbf{w}_1$ and $\mathbf{Q}^\top\mathbf{w}_n$, where $\mathbf{w}_1, \mathbf{w}_n$ are the first and last columns of \mathbf{W}^\top . \square

C.2 Approximating only the eigenvalues of a tridiagonal matrix

The following result gives the complexity of approximating only the eigenvalues of \mathbf{T} , instead of a full diagonalization. The main observation is that Lemma 2.1 can be simplified if we only need the eigenvalues. This simplification is listed in C.2 in the Appendix. Using this as an inductive step, we obtain the following Corollary.

Corollary C.1. *Let \mathbf{T} be a symmetric unreduced tridiagonal matrix with $\|\mathbf{T}\| \leq 1$ and $\epsilon \in (0, 1/2)$ be a given accuracy parameter. Assuming access to an (ϵ, n) -FMM, for $\tau \in \Theta(\text{poly}(\frac{\epsilon}{n}))$, we can compute approximate eigenvalues $\tilde{\lambda}_1, \dots, \tilde{\lambda}_n$ such that $|\tilde{\lambda}_i - \lambda_i(\mathbf{T})| \leq \epsilon$ in $O\left(n \log^{\xi+2}(\frac{n}{\epsilon})\right)$ arithmetic operations.*

Proof. The proof is similar to the one of Theorem 1.1. During the recursion, instead of Lemma 2.1, we use Lemma C.2. This way we reduce the arithmetic complexity of each recursive call from $O\left(n^2 \log^{\xi+1}(\frac{n}{\epsilon})\right)$ arithmetic operations to $O\left(n \log^{\xi+1}(\frac{n}{\epsilon})\right)$, which becomes $O\left(n \log^{\xi+2}(\frac{n}{\epsilon})\right)$ after solving the recursion. The approximation guarantees remain the same, but in this case we only compute the eigenvalues, without the eigenvectors. \square

C.3 Application: Hermitian diagonalization

Theorem 1.1 has a direct application to diagonalize Hermitian matrices. The following result is immediate.

Corollary C.2. *Let \mathbf{A} be a Hermitian matrix of size n with $\|\mathbf{A}\| \leq 1$. Given accuracy $\epsilon \in (0, 1/2)$, and an (ϵ, n) -FMM implementation of Prop. 1.1, we can compute a matrix $\tilde{\mathbf{Q}}$ and a diagonal matrix $\tilde{\Lambda}$ such that*

$$\left\| \mathbf{A} - \tilde{\mathbf{Q}} \tilde{\Lambda} \tilde{\mathbf{Q}}^* \right\| \leq \epsilon, \quad \left\| \tilde{\mathbf{Q}}^* \tilde{\mathbf{Q}} - \mathbf{I} \right\| \leq \epsilon/n^2,$$

or, stated alternatively,

$$\tilde{\mathbf{Q}} = \mathbf{Q} + \mathbf{E}_Q, \quad \mathbf{Q}^\top \mathbf{Q} = \mathbf{I}, \quad \|\mathbf{E}_Q\| \leq \epsilon/n^2, \quad \left\| \mathbf{A} - \mathbf{Q} \tilde{\Lambda} \mathbf{Q}^\top \right\| \leq \epsilon.$$

The algorithm requires a total of $O\left(n^\omega \log(n) + n^2 \log^{\xi+1}(\frac{n}{\epsilon})\right)$ arithmetic operations and comparisons.

Proof. Using the tridiagonal reduction algorithm of Schönhage [79], we can compute a unitary matrix \mathbf{Z} and a tridiagonal matrix \mathbf{T} in $O(n^\omega \log(n))$ arithmetic operations such that $\mathbf{A} = \mathbf{Z} \mathbf{T} \mathbf{Z}^*$ (see Theorem 3.1 for the complexity analysis). The matrix \mathbf{T} can always be chosen to have real entries. We then use Theorem 1.1 to compute $\tilde{\mathbf{U}}$ and $\tilde{\Lambda}$ such that

$$\left\| \mathbf{T} - \tilde{\mathbf{U}} \tilde{\Lambda} \tilde{\mathbf{U}}^\top \right\| \leq \epsilon, \quad \left\| \tilde{\mathbf{U}}^\top \tilde{\mathbf{U}} - \mathbf{I} \right\| \leq \epsilon/n^2,$$

in $O\left(n^2 \log^{\xi+1}(\frac{n}{\epsilon})\right)$ arithmetic operations. We then form $\tilde{\mathbf{Q}} = \mathbf{Z} \tilde{\mathbf{U}}$ in $O(n^\omega)$ operations. It holds that

$$\left\| \mathbf{A} - \tilde{\mathbf{Q}} \tilde{\Lambda} \tilde{\mathbf{Q}}^* \right\| = \left\| \mathbf{Z} \mathbf{T} \mathbf{Z}^* - \mathbf{Z} \tilde{\mathbf{U}} \tilde{\Lambda} \tilde{\mathbf{U}}^\top \mathbf{Z}^* \right\| = \left\| \mathbf{T} - \tilde{\mathbf{U}} \tilde{\Lambda} \tilde{\mathbf{U}}^\top \right\| \leq \epsilon,$$

and $\left\| \tilde{\mathbf{Q}}^* \tilde{\mathbf{Q}} - \mathbf{I} \right\| = \left\| \tilde{\mathbf{U}}^\top \mathbf{Z}^* \mathbf{Z} \tilde{\mathbf{U}} - \mathbf{I} \right\| \leq \epsilon/n^2$. The alternative statement can be obtained in a similar way as in the last part of the proof of Theorem 1.1. \square

C.4 Singular Value Decomposition

In this section we provide the proof of Theorem 1.2.

Proof of Theorem 1.2. To ensure that $\|\mathbf{A}\|$ satisfies the norm requirements, we first scale \mathbf{A} by $\frac{1}{\|\mathbf{A}\|_F} = \frac{1}{\sqrt{\text{tr}(\mathbf{A}^* \mathbf{A})}}$. The trace can be computed in $O(mn) = O(n^{k+1})$. Using standard trace and norm inequalities, it holds that

$$\text{tr}(\mathbf{A}^* \mathbf{A}) \leq n \|\mathbf{A}^\top \mathbf{A}\| = n \|\mathbf{A}\|^2 \leq n \|\mathbf{A}\|_F^2 = n \text{tr}(\mathbf{A}^* \mathbf{A}),$$

which gives the desired $\frac{1}{\sqrt{n}} \leq \|\frac{\mathbf{A}}{\sqrt{\text{tr}(\mathbf{A}^* \mathbf{A})}}\| \leq 1$.

The next step is to compute the Gramian $\mathbf{A}^* \mathbf{A}$ in $O(n^{\omega(1,k,1)})$ and then reduce it to tridiagonal form \mathbf{T} using the algorithm of [79] (see Theorem 3.1) in $O(n^\omega)$. We can now use Corollary C.1 to compute the condition number of \mathbf{T} . We start with $\epsilon = 1/2$, and call Corollary C.1 iteratively to compute the eigenvalues of \mathbf{T} up to error ϵ . We keep halving ϵ and calling in each iteration until $\tilde{\lambda}_{\min} > 0$ and $\epsilon \leq \frac{\tilde{\lambda}_{\min}}{4}$. This ensures that all eigenvalues are approximated up to additive error $\frac{1}{4} \lambda_{\min}(\mathbf{T}) = \frac{1}{4} \sigma_{\min}(\mathbf{T})$ (the equality holds because \mathbf{T} is PSD). Due to the assumption $\frac{1}{n^c} \leq \|\mathbf{A}\| \leq 1$, it holds that $\kappa(\mathbf{A}) \leq \frac{1}{\sigma_{\min}(\mathbf{A})} \leq n^c \kappa(\mathbf{A})$, in which case we can use $\tilde{\lambda}_{\min}$ to approximate the condition number of \mathbf{T} up to a factor n^c . In particular, if $\tilde{\sigma}_{\min} = \sqrt{\tilde{\lambda}_{\min}} \in \Theta(\sigma_{\min}(\mathbf{A}))$ is the approximated smallest singular value of \mathbf{T} , we can set $\tilde{\kappa} = \frac{1}{\tilde{\sigma}} \in \Theta(\frac{1}{\sigma_{\min}(\mathbf{A})}) \in [\Omega(\kappa(\mathbf{A})), O(n^c \kappa(\mathbf{A}))]$.

We call Corollary C.1 $O(\log(n\kappa(\mathbf{A})))$ times, and each call costs $O\left(n \log^{\xi+2}(\frac{n}{\epsilon})\right)$ arithmetic operations. The total cost is therefore at most $O\left(n \log^{\xi+3}(n\kappa(\mathbf{A}))\right)$.

Next, we set $\epsilon' = \epsilon/(n^c \tilde{\kappa})^2$ and use Corollary 2.1 compute a nearly unitary $\tilde{\mathbf{V}}$ and diagonal $\tilde{\Lambda}$ such that

$$\|\mathbf{A}^* \mathbf{A} - \tilde{\mathbf{V}} \tilde{\Lambda} \tilde{\mathbf{V}}^*\| \leq \epsilon', \quad \|\tilde{\mathbf{V}}^* \tilde{\mathbf{V}} - \mathbf{I}\| \leq \epsilon'/n^2.$$

This costs $O\left(n^\omega \log(n) + n^2 \log^{\xi+1}(\frac{n}{\epsilon'})\right) = O\left(n^\omega \log(n) + n^2 \log^{\xi+1}(\frac{n\kappa(\mathbf{A})}{\epsilon})\right)$. Due to the upper bound of ϵ' , the diagonal elements of $\tilde{\Lambda}$ are positive, and we set $\tilde{\Sigma} = \tilde{\Lambda}^{1/2}$.

The left singular vector matrix is set to $\tilde{\mathbf{U}} = \mathbf{A} \tilde{\mathbf{V}}^* \tilde{\Sigma}^{-1}$, which implies that $\mathbf{A} = \tilde{\mathbf{U}} \tilde{\Sigma} \tilde{\mathbf{V}}^*$. It remains to show that $\tilde{\mathbf{U}}$ has approximately orthonormal columns. We have that

$$\tilde{\mathbf{U}}^* \tilde{\mathbf{U}} = \tilde{\Sigma}^{-1} \tilde{\mathbf{V}}^{-1} \mathbf{A}^* \mathbf{A} \tilde{\mathbf{V}}^* \tilde{\Sigma}^{-1} = \tilde{\Sigma}^{-1} \tilde{\mathbf{V}}^{-1} (\tilde{\mathbf{V}} \tilde{\Lambda} \tilde{\mathbf{V}}^* + \mathbf{E}) \tilde{\mathbf{V}}^* \tilde{\Sigma}^{-1} = \mathbf{I} + \tilde{\Sigma}^{-1} \tilde{\mathbf{V}}^{-1} \mathbf{E} \tilde{\mathbf{V}}^* \tilde{\Sigma}^{-1},$$

where $\|\mathbf{E}\| \leq \epsilon'$. Then

$$\|\mathbf{I} - \tilde{\mathbf{U}}^* \tilde{\mathbf{U}}\| = \|\tilde{\Sigma}^{-1} \tilde{\mathbf{V}}^{-1} \mathbf{E} \tilde{\mathbf{V}}^* \tilde{\Sigma}^{-1}\| \leq \|\tilde{\Sigma}^{-1}\|^2 \|\tilde{\mathbf{V}}^{-1}\|^2 \|\mathbf{E}\| \leq C(n^c \kappa(\mathbf{A}))^2 \epsilon' \leq C\epsilon,$$

for some small constant C , which arises from $\|\tilde{\mathbf{V}}^{-1}\| \approx \frac{1}{1+\epsilon/\text{poly}(n)}$ and the unspecified constant factor of the approximate condition number $\tilde{\kappa}$, which should be also around 2 in the worst case. Rescaling ϵ' gives the final result.

For the alternative statement, we write $\tilde{\mathbf{U}} = \mathbf{U} + \mathbf{E}_U$, where $\mathbf{U}^* \mathbf{U} = \mathbf{I}$ and $\|\mathbf{E}_U\| \leq \epsilon$, and $\tilde{\mathbf{V}} = \mathbf{V} + \mathbf{E}_V$ where $\mathbf{V}^* \mathbf{V} = \mathbf{I}$ and $\|\mathbf{E}_V\| \ll \epsilon/n^2$. It is easy to see that $\|\tilde{\Sigma}\| \leq 1 + \epsilon$, $\|\tilde{\mathbf{U}}\| \leq 1 + \epsilon$, and $\|\tilde{\mathbf{V}}\| \ll 1 + \epsilon/n^2$. Then

$$\begin{aligned} \|\mathbf{A} - \mathbf{U} \tilde{\Sigma} \tilde{\mathbf{V}}^*\| &\leq \|\mathbf{A} - \tilde{\mathbf{U}} \tilde{\Sigma} \tilde{\mathbf{V}}^*\| + \|\tilde{\Sigma}\| (\|\mathbf{E}_U\| \|\tilde{\mathbf{V}}\| + \|\mathbf{E}_V\| \|\tilde{\mathbf{U}}\| + \|\mathbf{E}_U\| \|\mathbf{E}_V\|) \\ &\leq \epsilon + (1 + \epsilon) (\epsilon(1 + \epsilon/n^2) + \epsilon(1 + \epsilon)/n^2 + \epsilon^2/n^2) \\ &< 4\epsilon, \end{aligned}$$

and thus it suffices to rescale ϵ by $1/4$. □

D Floating point arithmetic

The sign s is + if the corresponding bit is one, and − if the bit is zero. The exponent e is stored as a binary number in the so-called *biased form*, and its range is $e \in [-M, M]$, where $M = 2^{p-1}$. The significand m is an integer that

satisfies $2^{t-1} \leq m \leq 2^t - 1$, where the lower bound is enforced to ensure that the system is *normalized*, i.e. the first bit of m is always 1. We can therefore write $\mathbf{fl}(\alpha)$ in a more intuitive representation

$$\mathbf{fl}(\alpha) = \pm 2^e \times \left(\frac{m_1}{2} + \frac{m_2}{2^2} + \dots + \frac{m_t}{2^t} \right),$$

where the first bit m_1 of m is always equal to one for normalized numbers. The range of normalized numbers is therefore $[2^{-M}, 2^M(2 - 2^{-t})]$. Numbers that are smaller than 2^{-M} are called *subnormal* and they will be ignored for simplicity, since we can either add more bits in the exponent. Similarly, numbers that are larger than $2^M(2 - 2^{-t})$ are assumed to be numerically equal to infinity, denoted by INF.

From [52, Theorem 2.2], for all real numbers α in the normalized range it holds that

$$\mathbf{fl}(\alpha) = (1 + \theta)\alpha,$$

where $\theta \in \mathbb{R}$ satisfies $|\theta| \leq 2^{-t} := \mathbf{u}$, where \mathbf{u} is the *machine precision*. Clearly, $t = O(\log(1/\mathbf{u}))$, in which case we can always obtain a bound for the number of required bits of a numerical algorithm if we have an upper bound for the precision \mathbf{u} . We will write the same for complex numbers which are represented as a pair of normalized floating point numbers.

The floating point implementation of each arithmetic operation $\odot \in \{+, -, \times, /\}$ also satisfies

$$\mathbf{fl}(\alpha \odot \beta) = (1 + \theta)(\alpha \odot \beta), \quad |\theta| \leq \mathbf{u}. \quad (34)$$

Divisions and multiplications with 1 and 2 do not introduce errors (for the latter we simply increase/decrease the exponent). We assume that we also have an implementation of $\sqrt{\cdot}$ such that $\mathbf{fl}(\sqrt{\alpha}) = (1 + \theta)\sqrt{\alpha}$ where $|\theta| \leq \mathbf{u}$. From [52, Lemma 3.1], we can bound products of errors as

$$\prod_{i=1}^n (1 + \theta_i)^{\rho_i} = 1 + \eta_n,$$

where $\rho_i = \pm 1$ and $|\eta_n| \leq \frac{n\mathbf{u}}{1 - n\mathbf{u}}$.

The above can be extended also for complex arithmetic (see [52, Lemma 3.5]), where the bound becomes $|\theta| \leq O(\mathbf{u})$, but we will ignore the constant prefactor for simplicity.

Operations on matrices can be analyzed in a similar manner. Let \otimes denote the element-wise multiplication between two matrices and \oslash the element-wise division. The floating point representation of a matrix \mathbf{A} satisfies

$$\mathbf{fl}(\mathbf{A}) = \mathbf{A} + \Delta \otimes \mathbf{A}, \quad |\Delta_{i,j}| \leq \mathbf{u}.$$

It can be shown that $\|\Delta\| \leq \mathbf{u}\sqrt{n}\|\mathbf{A}\|$. For any operation $\odot \in \{+, -, \otimes, \oslash\}$ and matrices \mathbf{A} and \mathbf{B} it holds that

$$\mathbf{fl}(\mathbf{A} \odot \mathbf{B}) = \mathbf{A} \odot \mathbf{B} + \Delta \otimes (\mathbf{A} \odot \mathbf{B}), \quad |\Delta_{i,j}| \leq \mathbf{u}, \quad \|\Delta \otimes (\mathbf{A} \odot \mathbf{B})\| \leq \mathbf{u}\sqrt{n}\|\mathbf{A} \odot \mathbf{B}\|. \quad (35)$$

E Reduction to tridiagonal form - omitted proofs and definitions

E.1 Imported subroutines

In this appendix we mention some preliminary results that we use in the analysis.

Theorem E.1 (MM, stable fast matrix multiplication [29, 28]). *For every $\eta > 0$, there exists a fast matrix multiplication algorithm MM which takes as input two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{n \times n}$ and returns $\mathbf{C} \leftarrow \text{MM}(\mathbf{A}, \mathbf{B})$ such that*

$$\|\mathbf{C} - \mathbf{AB}\| \leq n^{c_\eta} \cdot \mathbf{u} \|\mathbf{A}\| \|\mathbf{B}\|,$$

on floating point machine with precision \mathbf{u} , for some constant c_η independent of n . It requires $O(n^{\omega+\eta})$ floating point operations.

We can also assume that if the result \mathbf{AB} is Hermitian, then $\mathbf{MM}(\mathbf{A}, \mathbf{B})$ will be Hermitian as well (see e.g. [83]).

Theorem E.2 (Cf. [28]). *Given a matrix $\mathbf{A} \in \mathbb{C}^{m \times n}$, $m \geq n$, there exists an algorithm $[\mathbf{Q}, \mathbf{R}] \leftarrow \text{QR}(\mathbf{A})$ which returns an upper triangular matrix $\mathbf{R} \in \mathbb{C}^{m \times n}$ and a matrix $\mathbf{Q} \in \mathbb{C}^{m \times m}$, such that:*

$$(\mathbf{Q} + \mathbf{E}_Q)^*(\mathbf{A} + \mathbf{E}_A) = \mathbf{R}, \quad \|\mathbf{E}_Q\| \leq n^{c_{QR}} \mathbf{u}, \quad \|\mathbf{E}_A\| \leq n^{c_{QR}} \mathbf{u} \|\mathbf{A}\|,$$

for some constant c_{QR} , where the matrix $\mathbf{Q} + \mathbf{E}_Q$ is unitary. The algorithm executes $O(mn^{\omega-1})$ floating point operations on a machine with precision \mathbf{u} .

In our analysis it will be useful the following re-statement of the aforementioned results.

Corollary E.1. *There exists a global constant $\beta \geq \max\{c_{QR}, c_\eta, 1\}$, such that the matrices from Theorems E.1 and E.2 satisfy the following properties*

$$\begin{aligned} \|\mathbf{MM}(\mathbf{A}, \mathbf{B}) - \mathbf{AB}\| &\leq n^\beta \mathbf{u} \|\mathbf{A}\| \|\mathbf{B}\|, \\ \|\mathbf{A} - \mathbf{QR}\| &\leq n^\beta \mathbf{u} \|\mathbf{A}\|, \\ \|\mathbf{R}\| &\leq (1 + n^\beta \mathbf{u}) \|\mathbf{A}\|, \\ \|\mathbf{Q}\| &\leq \sqrt{1 + n^\beta \mathbf{u}}, \\ \|\mathbf{Q}^{-1}\| &\leq \frac{1}{\sqrt{1 - n^\beta \mathbf{u}}}, \\ \max\{\|\mathbf{I} - \mathbf{QQ}^*\|, \|\mathbf{I} - \mathbf{Q}^* \mathbf{Q}\|\} &\leq n^\beta \mathbf{u}. \end{aligned}$$

Proof. $\|\mathbf{R}\| = \|\mathbf{A} + \mathbf{E}_A\| \leq \|\mathbf{E}_A\| + \|\mathbf{A}\| \leq (1 + n^{c_{QR}} \mathbf{u}) \|\mathbf{A}\|$. Moreover, $\|\mathbf{Q}\| \leq \|\mathbf{Q} + \mathbf{E}_Q\| + \|\mathbf{E}_Q\| \leq 1 + n^{c_{QR}} \mathbf{u}$. Then from Theorem E.2 we can rewrite $\mathbf{A} + \mathbf{E}_A = (\mathbf{Q} + \mathbf{E}_Q)\mathbf{R}$. This gives

$$\begin{aligned} \|\mathbf{A} - \mathbf{QR}\| &= \|\mathbf{E}_Q \mathbf{R} - \mathbf{E}_A\| \\ &\leq \|\mathbf{E}_Q\| \|\mathbf{A} + \mathbf{E}_A\| + \|\mathbf{E}_A\| \\ &\leq n^{c_{QR}} \mathbf{u} (1 + n^{c_{QR}} \mathbf{u}) \|\mathbf{A}\| + n^{c_{QR}} \mathbf{u} \|\mathbf{A}\| \\ &= n^{c_{QR}} \mathbf{u} (2 + n^{c_{QR}} \mathbf{u}) \|\mathbf{A}\| \\ &\leq n^{2c_{QR}+1} \mathbf{u} \|\mathbf{A}\|, \end{aligned}$$

where in the last we assumed $n \geq 2$ and $\mathbf{u} \leq 1$. □

Using square fast matrix multiplication we can obtain an algorithm to compute two banded matrices with the same bandwidth

Corollary E.2. *Let \mathbf{A}, \mathbf{B} in $\mathbb{C}^{n \times n}$, where, without loss of generality, n is a power of two. Assume that \mathbf{A} and \mathbf{B} are block-tridiagonal, with block size $n_k = 2^k$ for some $k \in \{0, 1, \dots, \log(n) - 2\}$. We can compute a matrix \mathbf{C}' such that $\|\mathbf{C}' - \mathbf{AB}\| \leq \mathbf{u} n_k^\beta \|\mathbf{A}\| \|\mathbf{B}\|$ in $O(nn_k^{\omega-1})$ floating point operations.*

Proof. By definition, the true matrix $\mathbf{C} = \mathbf{AB}$ is block-pentadiagonal with the same block size as \mathbf{A} and \mathbf{B} . Each block of \mathbf{C} can be computed using at most three square matrix multiplications between blocks of \mathbf{A} and \mathbf{B} , and two square matrix additions. Therefore, each block $\mathbf{C}_{i,j}$ of \mathbf{C} can be computed in $O(n_k^\omega)$ floating point operations, and, by Theorem E.1 and Corollary E.1 the result will satisfy $\|\mathbf{C}'_{i,j} - \mathbf{C}_{i,j}\| \leq O(n_k^\beta \mathbf{u} \|\mathbf{A}\| \|\mathbf{B}\|)$. Splitting the matrix \mathbf{C} in three block-diagonals, \mathbf{C}_{-1} , \mathbf{C}_0 , and \mathbf{C}_1 , we have that

$$\|\mathbf{C}' - \mathbf{C}\| \leq \|\mathbf{C}_{-1} - \mathbf{C}'_{-1}\| + \|\mathbf{C}_0 - \mathbf{C}'_0\| + \|\mathbf{C}_1 - \mathbf{C}'_1\| \leq 3 \cdot O(n_k^\beta \mathbf{u} \|\mathbf{A}\| \|\mathbf{B}\|).$$

There are $3n/n_k - 2$ blocks in \mathbf{C} , which gives the total complexity to be $O(n_k^\omega n/n_k) = O(nn_k^{\omega-1})$. □

E.2 Rotations

An example of the rotations is illustrated in Equations (36) and (37). In Equation (36), the matrices $\mathbf{A}'_{1,2}$ and $\mathbf{A}'_{2,1}$ are lower and upper triangular, respectively. In Equation (37), the matrices $\mathbf{A}'_{4,2}$ and $\mathbf{A}'_{5,3}$ are upper triangular, while $\mathbf{A}'_{2,4}$ and $\mathbf{A}'_{3,5}$ are lower triangular.

$$\begin{array}{c} \overbrace{\begin{pmatrix} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} & \boxed{\mathbf{A}_{1,3}} & 0 & 0 & 0 & 0 & 0 \\ \mathbf{A}_{2,1} & \mathbf{A}_{2,2} & \mathbf{A}_{2,3} & \mathbf{A}_{2,4} & 0 & 0 & 0 & 0 \\ \boxed{\mathbf{A}_{3,1}} & \mathbf{A}_{3,2} & \mathbf{A}_{3,3} & \mathbf{A}_{3,4} & \mathbf{A}_{3,5} & 0 & 0 & 0 \\ 0 & \mathbf{A}_{4,2} & \mathbf{A}_{4,3} & \mathbf{A}_{4,4} & \mathbf{A}_{4,5} & \mathbf{A}_{4,6} & 0 & 0 \\ 0 & 0 & \mathbf{A}_{5,3} & \mathbf{A}_{5,4} & \mathbf{A}_{5,5} & \mathbf{A}_{5,6} & \mathbf{A}_{5,7} & 0 \\ 0 & 0 & 0 & \mathbf{A}_{6,4} & \mathbf{A}_{6,5} & \mathbf{A}_{6,6} & \mathbf{A}_{6,7} & \mathbf{A}_{6,8} \\ 0 & 0 & 0 & 0 & \mathbf{A}_{7,5} & \mathbf{A}_{7,6} & \mathbf{A}_{7,7} & \mathbf{A}_{7,8} \\ 0 & 0 & 0 & 0 & 0 & \mathbf{A}_{8,6} & \mathbf{A}_{8,7} & \mathbf{A}_{8,8} \end{pmatrix}}^{\mathbf{A}^{(k,2,0)}, i=2} \xrightarrow{R_2} \overbrace{\begin{pmatrix} \mathbf{A}_{1,1} & \mathbf{A}'_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{A}'_{2,1} & \mathbf{A}_{2,2} & \mathbf{A}_{2,3} & \mathbf{A}_{2,4} & \boxed{\mathbf{A}_{2,5}} & 0 & 0 & 0 \\ 0 & \mathbf{A}_{3,2} & \mathbf{A}_{3,3} & \mathbf{A}_{3,4} & \mathbf{A}_{3,5} & 0 & 0 & 0 \\ 0 & \mathbf{A}_{4,2} & \mathbf{A}_{4,3} & \mathbf{A}_{4,4} & \mathbf{A}_{4,5} & \mathbf{A}_{4,6} & 0 & 0 \\ 0 & \boxed{\mathbf{A}_{5,2}} & \mathbf{A}_{5,3} & \mathbf{A}_{5,4} & \mathbf{A}_{5,5} & \mathbf{A}_{5,6} & \mathbf{A}_{5,7} & 0 \\ 0 & 0 & 0 & \mathbf{A}_{6,4} & \mathbf{A}_{6,5} & \mathbf{A}_{6,6} & \mathbf{A}_{6,7} & \mathbf{A}_{6,8} \\ 0 & 0 & 0 & 0 & \mathbf{A}_{7,5} & \mathbf{A}_{7,6} & \mathbf{A}_{7,7} & \mathbf{A}_{7,8} \\ 0 & 0 & 0 & 0 & 0 & \mathbf{A}_{8,6} & \mathbf{A}_{8,7} & \mathbf{A}_{8,8} \end{pmatrix}}^{\mathbf{A}^{(k,3,5)}} \end{array} \quad (36)$$

$$\begin{array}{c} \overbrace{\begin{pmatrix} \mathbf{A}_{1,1} & \mathbf{A}'_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{A}'_{2,1} & \mathbf{A}_{2,2} & \mathbf{A}_{2,3} & \mathbf{A}_{2,4} & \boxed{\mathbf{A}_{2,5}} & 0 & 0 & 0 \\ 0 & \mathbf{A}_{3,2} & \mathbf{A}_{3,3} & \mathbf{A}_{3,4} & \mathbf{A}_{3,5} & 0 & 0 & 0 \\ 0 & \mathbf{A}_{4,2} & \mathbf{A}_{4,3} & \mathbf{A}_{4,4} & \mathbf{A}_{4,5} & \mathbf{A}_{4,6} & 0 & 0 \\ 0 & \boxed{\mathbf{A}_{5,2}} & \mathbf{A}_{5,3} & \mathbf{A}_{5,4} & \mathbf{A}_{5,5} & \mathbf{A}_{5,6} & \mathbf{A}_{5,7} & 0 \\ 0 & 0 & 0 & \mathbf{A}_{6,4} & \mathbf{A}_{6,5} & \mathbf{A}_{6,6} & \mathbf{A}_{6,7} & \mathbf{A}_{6,8} \\ 0 & 0 & 0 & 0 & \mathbf{A}_{7,5} & \mathbf{A}_{7,6} & \mathbf{A}_{7,7} & \mathbf{A}_{7,8} \\ 0 & 0 & 0 & 0 & 0 & \mathbf{A}_{8,6} & \mathbf{A}_{8,7} & \mathbf{A}_{8,8} \end{pmatrix}}^{\mathbf{A}^{(k,3,5)}} \xrightarrow{R'_4} \overbrace{\begin{pmatrix} \mathbf{A}_{1,1} & \mathbf{A}'_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{A}'_{2,1} & \mathbf{A}_{2,2} & \mathbf{A}_{2,3} & \mathbf{A}'_{2,4} & 0 & 0 & 0 & 0 \\ 0 & \mathbf{A}_{3,2} & \mathbf{A}_{3,3} & \mathbf{A}_{3,4} & \mathbf{A}'_{3,5} & 0 & 0 & 0 \\ 0 & \mathbf{A}'_{4,2} & \mathbf{A}_{4,3} & \mathbf{A}_{4,4} & \mathbf{A}_{4,5} & \mathbf{A}_{4,6} & \boxed{\mathbf{A}_{4,7}} & 0 \\ 0 & 0 & \mathbf{A}'_{5,3} & \mathbf{A}_{5,4} & \mathbf{A}_{5,5} & \mathbf{A}_{5,6} & \mathbf{A}_{5,7} & 0 \\ 0 & 0 & 0 & \mathbf{A}_{6,4} & \mathbf{A}_{6,5} & \mathbf{A}_{6,6} & \mathbf{A}_{6,7} & \mathbf{A}_{6,8} \\ 0 & 0 & 0 & \boxed{\mathbf{A}_{4,7}} & \mathbf{A}_{7,5} & \mathbf{A}_{7,6} & \mathbf{A}_{7,7} & \mathbf{A}_{7,8} \\ 0 & 0 & 0 & 0 & 0 & \mathbf{A}_{8,6} & \mathbf{A}_{8,7} & \mathbf{A}_{8,8} \end{pmatrix}}^{\mathbf{A}^{(k,3,7)}} \end{array} \quad (37)$$

The rotations can be conveniently implemented in floating point using fast QR factorizations.

Lemma E.1. Let $\mathbf{A}^{(k,i,0)}$ be a block-pentadiagonal matrix with no bulge of the form of Equation (4), consisting of $b_k \times b_k$ blocks of size $n_k \times n_k = 2^k \times 2^k$ each. The rotation $[\mathbf{A}^{(k,i+1,i+3)}, \mathbf{Q}] \leftarrow R_i(\mathbf{A}^{(k,i,0)})$ can be implemented in $O(n_k^\omega)$ floating point operations. The resulting matrix $\mathbf{A}^{(k,i+1,i+3)}$ has a bulge at positions $(i, i+3)$ and $(i+3, i)$, and \mathbf{Q} is approximately unitary. Moreover, if $\mathbf{u} \leq 1/n^\beta$, then

$$\|\mathbf{A}^{(k,i,0)} - \mathbf{Q}\mathbf{A}^{(k,i+1,i+3)}\mathbf{Q}^*\| \leq C\mathbf{u}n^\beta \|\mathbf{A}^{(k,i,0)}\|,$$

where C is a constant independent of n .

Proof. The first step is to compute $\mathbf{Q}_i, \mathbf{R}_i \leftarrow \text{QR}(\mathbf{A}_i)$, where $\mathbf{A}_i = \begin{pmatrix} \mathbf{A}_{i,i-1} \\ \mathbf{A}_{i+1,i-1} \end{pmatrix}$. From Theorem E.2, \mathbf{Q}_i has size $2n_k \times 2n_k$, and \mathbf{R}_i has size $2n_k \times n_k$ and it is upper triangular.

We then isolate the matrices $\mathbf{B}_i = \begin{pmatrix} \mathbf{A}_{i,i} & \mathbf{A}_{i,i+1} \\ \mathbf{A}_{i+1,i} & \mathbf{A}_{i+1,i+1} \end{pmatrix}$, as well as $\mathbf{C}_i = \begin{pmatrix} \mathbf{A}_{i,i+2} & 0 \\ \mathbf{A}_{i+1,i+2} & \mathbf{A}_{i+1,i+3} \end{pmatrix}$, and compute the two matrix products $\mathbf{B}'_i = \text{MM}(\mathbf{Q}_i^*, \mathbf{B}_i)$ and $\mathbf{C}'_i = \text{MM}(\mathbf{Q}_i^*, \mathbf{C}_i)$. We finally compute the product $\mathbf{B}''_i = \text{MM}(\mathbf{B}'_i, \mathbf{Q}_i)$. The matrix \mathbf{B}''_i can be forced to be Hermitian by copying the lower triangular part to the upper triangular part. This increases the error norm by a factor of $O(\log(n_k))$ (see eg [83]). Now consider the matrix $\mathbf{A}^{(k,i+1,i+3)}$, which is exactly the same as the original $\mathbf{A}^{(k,i,0)}$, having only replaced the following blocks:

$$\begin{pmatrix} \dots & \mathbf{A}_i^* & 0 \\ \mathbf{A}_i & \mathbf{B}_i & \mathbf{C}_i \\ 0 & \mathbf{C}_i^* & \dots \end{pmatrix} \longrightarrow \begin{pmatrix} \dots & \mathbf{R}_i^* & 0 \\ \mathbf{R}_i & \mathbf{B}''_i & \mathbf{C}'_i \\ 0 & \mathbf{C}_i'^* & \dots \end{pmatrix}.$$

If we set $\mathbf{Q} = \begin{pmatrix} \mathbf{I}_{(i-1)n_k} & 0 & 0 \\ 0 & \mathbf{Q}_i & 0 \\ 0 & 0 & \mathbf{I}_{n-(i+1)n_k} \end{pmatrix}$, then

$$\begin{aligned} \left\| \mathbf{A}^{(k,i,0)} - \mathbf{Q} \mathbf{A}^{(k,i+1,i+3)} \mathbf{Q}^* \right\| &= \left\| \begin{pmatrix} \cdots & \mathbf{A}_i^* & 0 \\ \mathbf{A}_i & \mathbf{B}_i & \mathbf{C}_i \\ 0 & \mathbf{C}_i^* & \cdots \end{pmatrix} - \begin{pmatrix} \cdots & \mathbf{R}_i^* \mathbf{Q}_i^* & 0 \\ \mathbf{Q}_i \mathbf{R}_i & \mathbf{Q}_i \mathbf{B}_i'' \mathbf{Q}_i^* & \mathbf{Q}_i \mathbf{C}_i' \\ 0 & \mathbf{C}_i' \mathbf{Q}_i^* & \cdots \end{pmatrix} \right\| \\ &\leq \left\| \mathbf{A}_i - \mathbf{Q}_i \mathbf{R}_i \right\| + \left\| \mathbf{B}_i - \mathbf{Q}_i \mathbf{B}_i'' \mathbf{Q}_i^* \right\| + \left\| \mathbf{C}_i - \mathbf{Q}_i \mathbf{C}_i' \right\|. \end{aligned} \quad (38)$$

From Corollary E.1 we have that $\left\| \mathbf{A}_i - \mathbf{Q}_i \mathbf{R}_i \right\| \leq (2n_k)^\beta \mathbf{u} \left\| \mathbf{A}_i \right\|$. For the middle term, note that we can write $\mathbf{B}_i'' = (\mathbf{Q}_i^* \mathbf{B}_i + \mathbf{E}_1) \mathbf{Q}_i + \mathbf{E}_2$. From Corollary E.1, $\left\| \mathbf{E}_1 \right\| \leq (2n_k)^\beta \mathbf{u} \left\| \mathbf{Q}_i \right\| \left\| \mathbf{B}_i \right\| \leq (2n_k)^\beta \mathbf{u} (1 + (2n_k)^\beta \mathbf{u}) \left\| \mathbf{B}_i \right\|$, and

$$\begin{aligned} \left\| \mathbf{E}_2 \right\| &\leq (2n_k)^\beta \mathbf{u} \left\| \mathbf{Q}_i^* \mathbf{B}_i + \mathbf{E}_1 \right\| \left\| \mathbf{Q}_i \right\| \\ &\leq (2n_k)^\beta \mathbf{u} \left((1 + (2n_k)^\beta \mathbf{u}) \left\| \mathbf{B}_i \right\| + (2n_k)^\beta \mathbf{u} (1 + (2n_k)^\beta \mathbf{u}) \left\| \mathbf{B}_i \right\| \right) (1 + (2n_k)^\beta \mathbf{u}) \\ &\leq (2n_k)^\beta \mathbf{u} \left(1 + (2n_k)^\beta \mathbf{u} \right)^3 \left\| \mathbf{B}_i \right\|. \end{aligned}$$

From Corollary E.1, we can write $\mathbf{Q} \mathbf{Q}^* = \mathbf{I} + \mathbf{E}_3$ where $\left\| \mathbf{E}_3 \right\| \leq (2n_k)^\beta \mathbf{u}$. The matrix $\mathbf{Q}_i \mathbf{B}_i'' \mathbf{Q}_i^*$ can be written as

$$\begin{aligned} \mathbf{Q}_i \mathbf{B}_i'' \mathbf{Q}_i^* &= \mathbf{Q}_i ((\mathbf{Q}_i^* \mathbf{B}_i + \mathbf{E}_1) \mathbf{Q}_i + \mathbf{E}_2) \mathbf{Q}_i^* \\ &= \mathbf{Q}_i \mathbf{Q}_i^* \mathbf{B}_i \mathbf{Q}_i \mathbf{Q}_i^* + \mathbf{Q}_i \mathbf{E}_1 \mathbf{Q}_i \mathbf{Q}_i^* + \mathbf{Q}_i \mathbf{E}_2 \mathbf{Q}_i^* \\ &= (\mathbf{I} + \mathbf{E}_3) \mathbf{B}_i (\mathbf{I} + \mathbf{E}_3) + \mathbf{Q}_i \mathbf{E}_1 (\mathbf{I} + \mathbf{E}_3) + \mathbf{Q}_i \mathbf{E}_2 \mathbf{Q}_i^* \\ &= \mathbf{B}_i + \mathbf{E}_3 \mathbf{B}_i + \mathbf{B}_i \mathbf{E}_3 + \mathbf{E}_3 \mathbf{B}_i \mathbf{E}_3 + \mathbf{Q}_i \mathbf{E}_1 + \mathbf{Q}_i \mathbf{E}_1 \mathbf{E}_3 + \mathbf{Q}_i \mathbf{E}_2 \mathbf{Q}_i^*. \end{aligned}$$

Then

$$\begin{aligned} \left\| \mathbf{B}_i - \mathbf{Q}_i \mathbf{B}_i'' \mathbf{Q}_i^* \right\| &= \left\| \mathbf{E}_3 \mathbf{B}_i + \mathbf{B}_i \mathbf{E}_3 + \mathbf{E}_3^2 + \mathbf{Q}_i \mathbf{E}_1 + \mathbf{Q}_i \mathbf{E}_1 \mathbf{E}_3 + \mathbf{Q}_i \mathbf{E}_2 \mathbf{Q}_i^* \right\| \\ &\leq 2 \left\| \mathbf{E}_3 \mathbf{B}_i \right\| + \left\| \mathbf{E}_3 \mathbf{B}_i \mathbf{E}_3 \right\| + \left\| \mathbf{Q}_i \mathbf{E}_1 \right\| + \left\| \mathbf{Q}_i \mathbf{E}_1 \mathbf{E}_3 \right\| + \left\| \mathbf{Q}_i \mathbf{E}_2 \mathbf{Q}_i^* \right\| \\ &\leq 2(2n_k)^\beta \mathbf{u} \left\| \mathbf{B}_i \right\| + ((2n_k)^\beta \mathbf{u})^2 \left\| \mathbf{B}_i \right\| + (2n_k)^\beta \mathbf{u} (1 + (2n_k)^\beta \mathbf{u})^2 \left\| \mathbf{B}_i \right\| + \dots \\ &\quad \dots ((2n_k)^\beta \mathbf{u})^2 (1 + (2n_k)^\beta \mathbf{u})^2 \left\| \mathbf{B}_i \right\| + (2n_k)^\beta \mathbf{u} \left(1 + (2n_k)^\beta \mathbf{u} \right)^5 \left\| \mathbf{B}_i \right\| \\ &\leq (2n_k)^\beta \mathbf{u} \left\| \mathbf{B}_i \right\| \left(2 + (2n_k)^\beta \mathbf{u} + (1 + (2n_k)^\beta \mathbf{u})^2 + (2n_k)^\beta \mathbf{u} (1 + (2n_k)^\beta \mathbf{u})^2 + \left(1 + (2n_k)^\beta \mathbf{u} \right)^5 \right) \\ &\leq C_1 (2n_k)^\beta \mathbf{u} \left\| \mathbf{B}_i \right\|, \end{aligned}$$

for some constant C_1 , where $\mathbf{u} \leq 1/n^\beta$ is a sufficient condition for the last inequality.

Finally, note that from Corollary E.1 \mathbf{C}_i' can be written as $\mathbf{C}_i' = (\mathbf{Q}_i^* \mathbf{C}_i + \mathbf{E}_4)$ where $\left\| \mathbf{E}_4 \right\| \leq (2n_k)^\beta \mathbf{u} \left\| \mathbf{Q}_i \right\| \left\| \mathbf{C}_i \right\| \leq (2n_k)^\beta \mathbf{u} (1 + (2n_k)^\beta \mathbf{u}) \left\| \mathbf{C}_i \right\|$, in which case

$$\begin{aligned} \left\| \mathbf{C}_i - \mathbf{Q}_i \mathbf{C}_i' \right\| &= \left\| \mathbf{C}_i - \mathbf{Q}_i (\mathbf{Q}_i^* \mathbf{C}_i + \mathbf{E}_4) \right\| \\ &\leq \left\| \mathbf{C}_i - (\mathbf{I} + \mathbf{E}_3) \mathbf{C}_i \right\| + \left\| \mathbf{Q}_i \mathbf{E}_4 \right\| \\ &\leq \left\| \mathbf{E}_3 \mathbf{C}_i \right\| + \left\| \mathbf{Q}_i \mathbf{E}_4 \right\| \\ &\leq (2n_k)^\beta \mathbf{u} \left\| \mathbf{C}_i \right\| + (2n_k)^\beta \mathbf{u} (1 + (2n_k)^\beta \mathbf{u})^2 \left\| \mathbf{C}_i \right\| \\ &= (2n_k)^\beta \mathbf{u} \left\| \mathbf{C}_i \right\| (1 + (2n_k)^\beta \mathbf{u} (1 + (2n_k)^\beta \mathbf{u})^2) \\ &\leq C_2 (2n_k)^\beta \mathbf{u} \left\| \mathbf{C}_i \right\|, \end{aligned}$$

where C_2 is a constant which arises if we assume that $\mathbf{u} \leq 1/(2n_k)^\beta$. Replacing all the derived bounds in Eq. (38) it follows that

$$\left\| \mathbf{A}^{(k,i,0)} - \mathbf{Q} \mathbf{A}^{(k,i+1,i+3)} \mathbf{Q}^* \right\| \leq (2n_k)^\beta \mathbf{u} (\left\| \mathbf{A}_i \right\| + C_1 \left\| \mathbf{B}_i \right\| + C_2 \left\| \mathbf{C}_i \right\|) \leq C_3 (2n_k)^\beta \mathbf{u} \left\| \mathbf{A}^{(k,i,0)} \right\|,$$

where we bounded the norms of the submatrices by the norm of the original matrix and $C_3 = 1 + C_1 + C_2$. \square

Lemma E.2. Let $\mathbf{A}^{(k,s,j+1)}$, $j \geq s+1$, be a block-pentadiagonal matrix of the form of Equation (4), with a bulge at position $(j+1, j-2)$, consisting of $b_k \times b_k$ blocks of size $n_k \times n_k = 2^k \times 2^k$ each. The rotation $[\mathbf{A}^{(k,s,j+3)}, \mathbf{Q}] \leftarrow R'_j(\mathbf{A}^{(k,s,j+1)})$ can be implemented in $O(n_k^\omega)$ floating point operations. The resulting matrix $\mathbf{A}^{(k,s,j+3)}$ has a bulge at positions $(j+3, j)$ and $(j, j+3)$, and \mathbf{Q} is approximately unitary. Moreover, if $\mathbf{u} \leq 1/n^\beta$, then

$$\|\mathbf{A}^{(k,s,j+1)} - \mathbf{Q}\mathbf{A}^{(k,s,j+3)}\mathbf{Q}^*\| \leq C' \mathbf{u} (2n_k)^\beta \|\mathbf{A}^{(k,s,j+1)}\|,$$

where C' is a constant independent of n .

Proof. The proof is almost identical to the one of Lemma E.1 and therefore it is omitted. The only change is that instead of computing the QR of a $2n_k \times n_k$ submatrix, we compute the QR of a $2n_k \times 2n_k$ one. \square

E.3 Bandwidth halving

Lemma E.3. Let $\mathbf{A}^{(k,2,0)}$ be a full block-pentadiagonal matrix of the form of Equation (4), with no bulge, consisting of $b_k \times b_k$ blocks of size $n_k \times n_k = 2^k \times 2^k$ each. For any $k \in [\log(n) - 2]$, if $\mathbf{u} \leq \frac{1}{C_1 n^{\beta+3}}$, where C_1 is a constant, then

Algorithm 2 returns a matrix $\widehat{\mathbf{Q}}^{(k)}$ that is approximately orthogonal, and a matrix $\mathbf{A}^{(k-1,2,0)}$ such that

$$\begin{aligned} \|\mathbf{A}^{(k,2,0)} - \widehat{\mathbf{Q}}^{(k)} \mathbf{A}^{(k-1,2,0)} \widehat{\mathbf{Q}}^{(k)*}\| &\leq C_2 \mathbf{u} n^{\beta+3} \|\mathbf{A}^{(k,2,0)}\|, \\ \|\widehat{\mathbf{Q}}^{(k)} \widehat{\mathbf{Q}}^{(k)*} - \mathbf{I}\| &\leq C_3 \mathbf{u} n^{\beta+3} \end{aligned}$$

where C_2, C_3 are also constants independent of n . It requires $O(n^2 (S_\omega(\log(n) - 1) - S_\omega(k)))$ floating point operations, where $S_x(m) := \sum_{l=1}^m (2^{x-2})^l$. If $\widehat{\mathbf{Q}}^{(k)}$ is not required, the complexity reduces to $O(n^2 n_k^{\omega-2}) = O(n^2 (2^{\omega-2})^k)$.

Proof. The first observation is that, for fixed i (i.e. for every outer iteration), the rotation matrices $\mathbf{Q}_{i,j}$ for $j = i+1, \dots, b_k$ are all independent. This means that the product can be formed without any multiplications. Specifically, for even i , the product $\mathbf{Q}_i = \mathbf{Q}_{i,b_k} \cdot \mathbf{Q}_{i,b_k-2} \cdot \dots \cdot \mathbf{Q}_{i,i+2} \cdot \mathbf{Q}_{i,i}$ has the form

$$\mathbf{Q}_i = \begin{pmatrix} \mathbf{I} & & & & & \\ & \ddots & & & & \\ & & \mathbf{I} & & & \\ & & & \mathbf{Z}_{i,i} & & \\ & & & & \mathbf{Z}_{i,i+2} & \\ & & & & & \ddots \\ & & & & & & \mathbf{Z}_{i,b_k} \end{pmatrix}, \quad (39)$$

where the matrices \mathbf{Z} have size $2n_k \times 2n_k$ and originate from the QR factorizations inside the rotations R_i and R'_j . For odd i , the matrix $\mathbf{Q}_i = \mathbf{Q}_{i,b_k-1} \cdot \mathbf{Q}_{i,b_k-3} \cdot \dots \cdot \mathbf{Q}_{i,i+2} \cdot \mathbf{Q}_{i,i}$ has a similar form.

Using Lemmas E.1 and E.2, for some fixed even i , we can write

$$\begin{aligned} \mathbf{A}^{(k,i,0)} &= \mathbf{Q}_{i,i} \mathbf{A}^{(k,i+1,i+3)} \mathbf{Q}_{i,i}^* + \mathbf{E}_{i,i} \\ &= \mathbf{Q}_{i,i} \left(\mathbf{Q}_{i,i+2} \mathbf{A}^{(k,i+1,i+5)} \mathbf{Q}_{i,i+2}^* + \mathbf{E}_{i,i+2} \right) \mathbf{Q}_{i,i}^* + \mathbf{E}_{i,i} \\ &= \mathbf{Q}_i \mathbf{A}^{(k,i+1,0)} \mathbf{Q}_i^* + \mathbf{E}_{i,i} + \sum_{l=1}^{b_k/2} \left[\left(\prod_{m=1}^l \mathbf{Q}_{i,i+2m} \right) \mathbf{E}_{i,i+2l} \left(\prod_{m=1}^l \mathbf{Q}_{i,i+2m} \right)^* \right]. \end{aligned}$$

Here error matrix $\mathbf{E}_{i,i}$ originates from the first rotation R_i and is equal to $\mathbf{E}_{i,i} = \mathbf{A}^{(k,i,0)} - \mathbf{Q}_{i,i} \mathbf{A}^{(k,i+1,i+3)} \mathbf{Q}_{i,i}^*$, while for $l = 1, \dots, b_k/2$ the error matrices originate from the rotations R' and are equal to $\mathbf{E}_{i,i+2l} = \mathbf{A}^{(k,i+1,i+1+2l)} - \mathbf{Q}_{i,i+2l} \mathbf{A}^{(k,i+1,i+1+4l)} \mathbf{Q}_{i,i+2l}^*$. It holds that

$$\|\mathbf{E}_{i,i+2l}\| \leq C' \mathbf{u} (2n_k)^\beta \|\mathbf{A}^{(k,i+1,i+1+2l)}\| \leq C' \mathbf{u} n^\beta \|\mathbf{A}^{(k,i+1,i+1+2l)}\|.$$

From Lemma E.1 it also holds that $\|\mathbf{E}_{i,i}\| \leq C\mathbf{u}(2n_k)^\beta \|\mathbf{A}^{(k,i,0)}\| \leq C\mathbf{u}n^\beta \|\mathbf{A}^{(k,i,0)}\|$, and Corollary E.1 implies that the matrices $\mathbf{Q}_{i,i+2l}$ are invertible and $\|\mathbf{Q}_{i,i+2l}^{-1}\| = \frac{1}{\sigma_{\min}(\mathbf{Q}_{i,i+2l})} \geq \frac{1}{\sqrt{1-(2n_k)^\beta \mathbf{u}}} \geq \frac{1}{\sqrt{1-n^\beta \mathbf{u}}}$. This provides that

$$\|\mathbf{A}^{(k,i+1,i+1+4l)}\| = \|\mathbf{Q}_{i,i+2l}^{-1}(\mathbf{A}^{(k,i+1,i+1+2l)} - \mathbf{E}_{i,i+2l})\mathbf{Q}_{i,i+2l}^{-*}\| \leq \frac{1+C'n^\beta \mathbf{u}}{1-n^\beta \mathbf{u}} \|\mathbf{A}^{(k,i+1,i+1+2l)}\|,$$

and accordingly $\|\mathbf{A}^{(k,i+1,i+3)}\| \leq \frac{1+Cn^\beta \mathbf{u}}{1-n^\beta \mathbf{u}} \|\mathbf{A}^{(k,i,0)}\|$. Since, as already mentioned, the product $\prod_{m=1}^l \mathbf{Q}_{i,i+2m}$ involves independent diagonal blocks, then for any l :

$$\left\| \prod_{m=1}^l \mathbf{Q}_{i,i+2m} \right\| = \max_{m=1,\dots,l} \{\|\mathbf{Q}_{i,i+2m}\|\} \leq \sqrt{1+n^\beta \mathbf{u}}.$$

where the last comes from Corollary E.1. Similarly,

$$\left\| \left(\prod_{m=1}^l \mathbf{Q}_{i,i+2m} \right)^{-1} \right\| = \max_{m=1,\dots,l} \{\|(\mathbf{Q}_{i,i+2m})^{-1}\|\} \leq \frac{1}{\sqrt{1-n^\beta \mathbf{u}}}.$$

Combining the aforementioned observations we obtain

$$\begin{aligned} \|\mathbf{A}^{(k,i,0)} - \mathbf{Q}_i \mathbf{A}^{(k,i+1,0)} \mathbf{Q}_i^*\| &= \left\| \mathbf{E}_{i,i} + \sum_{l=1}^{b_k/2} \left[\left(\prod_{m=1}^l \mathbf{Q}_{i,i+2m} \right) \mathbf{E}_{i,i+2l} \left(\prod_{m=1}^l \mathbf{Q}_{i,i+2m} \right)^* \right] \right\| \\ &\leq \|\mathbf{E}_{i,i}\| + \sum_{l=1}^{b_k/2} \left[\|\mathbf{E}_{i,i+2l}\| \left\| \prod_{m=1}^l \mathbf{Q}_{i,i+2m} \right\|^2 \right] \\ &\leq \|\mathbf{E}_{i,i}\| + \left(\sqrt{1+n^\beta \mathbf{u}} \right)^2 \sum_{l=1}^{b_k/2} \|\mathbf{E}_{i,i+2l}\|. \\ &\leq C\mathbf{u}n^\beta \|\mathbf{A}^{(k,i,0)}\| + \left(1+n^\beta \mathbf{u} \right) \sum_{l=1}^{b_k/2} C'\mathbf{u}n^\beta \|\mathbf{A}^{(k,i+1,i+1+2l)}\|. \\ &\leq \mathbf{u}n^\beta \left[C\|\mathbf{A}^{(k,i,0)}\| + C' \left(1+n^\beta \mathbf{u} \right) \|\mathbf{A}^{(k,i+1,i+3)}\| \sum_{l=1}^{b_k/2} \left(\frac{1+C'n^\beta \mathbf{u}}{1-n^\beta \mathbf{u}} \right)^l \right] \\ &\leq \mathbf{u}n^\beta \left[C\|\mathbf{A}^{(k,i,0)}\| + C' \left(1+n^\beta \mathbf{u} \right) \frac{1+Cn^\beta \mathbf{u}}{1-n^\beta \mathbf{u}} \|\mathbf{A}^{(k,i,0)}\| \sum_{l=1}^{b_k/2} \left(\frac{1+C'n^\beta \mathbf{u}}{1-n^\beta \mathbf{u}} \right)^l \right] \\ &= \mathbf{u}n^\beta \|\mathbf{A}^{(k,i,0)}\| \left[C + C' \left(1+n^\beta \mathbf{u} \right) \frac{1+Cn^\beta \mathbf{u}}{1-n^\beta \mathbf{u}} \sum_{l=1}^{b_k/2} \left(\frac{1+C'n^\beta \mathbf{u}}{1-n^\beta \mathbf{u}} \right)^l \right]. \end{aligned}$$

If we further assume that $\mathbf{u} \leq \frac{1}{\max\{1, C, C'\} n^{\beta+1}}$, then all the summands are bounded by constants, and $b_k \leq n$, in which case we conclude

$$\|\mathbf{A}^{(k,i,0)} - \mathbf{Q}_i \mathbf{A}^{(k,i+1,0)} \mathbf{Q}_i^*\| \leq \mathbf{u}n^\beta \cdot C_1 n \|\mathbf{A}^{(k,i,0)}\|, \quad (40)$$

for some constant C_1 . Moreover, the matrix \mathbf{Q}_i is approximately orthogonal, since, from Eq. (39), it consists of approximately orthogonal diagonal blocks. In particular,

$$\|\mathbf{Q}_i \mathbf{Q}_i^* - \mathbf{I}\| \leq \max_{j=i+2, \dots, b_k-1} \{\|\mathbf{Z}_{i,j} \mathbf{Z}_{i,j}^* - \mathbf{I}\|\} \leq n^\beta \mathbf{u}. \quad (41)$$

We can now analyze the total error after applying the previous steps for all $i \in 2, \dots, b_k - 1$. From Eq. (40), if we denote by $\mathbf{E}_i = \mathbf{A}^{(k,i,0)} - \mathbf{Q}_i \mathbf{A}^{(k,i+1,0)} \mathbf{Q}_i^*$, we can directly obtain

$$\left\| \mathbf{A}^{(k,i+1,0)} \right\| = \left\| \mathbf{Q}_i^{-1} \left(\mathbf{A}^{(k,i,0)} - \mathbf{E}_i \right) \mathbf{Q}_i^{-*} \right\| \leq \left\| \mathbf{Q}_i^{-1} \right\|^2 \left(\left\| \mathbf{A}^{(k,i,0)} \right\| + \left\| \mathbf{E}_i \right\| \right) \leq \frac{1+\mathbf{u}n^\beta C_1 n}{1-n^\beta \mathbf{u}} \left\| \mathbf{A}^{(k,i,0)} \right\|.$$

We now denote $\mathbf{Q}^{(k)} = \mathbf{Q}_{b_k} \cdot \mathbf{Q}_{b_k-1} \cdot \dots \cdot \mathbf{Q}_3 \cdot \mathbf{Q}_2$. For now this product is assumed in exact arithmetic. We will later show how to efficiently compute it in floating point. In exact arithmetic, for any $l \in \{2, \dots, b_k\}$:

$$\left\| \prod_{m=2}^l \mathbf{Q}_m \right\| \leq \left(\sqrt{1+n^\beta \mathbf{u}} \right)^{l-1}, \quad \text{and} \quad \left\| \left(\prod_{m=2}^l \mathbf{Q}_m \right)^{-1} \right\| \leq \left(\frac{1}{\sqrt{1-n^\beta \mathbf{u}}} \right)^{l-1}.$$

From Bernoulli's inequalities, since $n^\beta \mathbf{u} \leq 1$, it follows that

$$\begin{aligned} \sigma_{\max}(\mathbf{Q}^{(k)})^2 &= \left\| \mathbf{Q}^{(k)} \right\|^2 \leq (1+n^\beta \mathbf{u})^{\log(n)-1} \leq 1+n^{\beta+1} \mathbf{u}, \\ \sigma_{\min}(\mathbf{Q}^{(k)})^2 &= \left\| \left(\mathbf{Q}^{(k)} \right)^{-1} \right\|^{-2} \geq (1-n^\beta \mathbf{u})^{(\log(n)-1)} \geq 1-(\log(n)-1)n^\beta \mathbf{u} \geq 1-n^{\beta+1} \mathbf{u}, \end{aligned}$$

and

$$\left\| \mathbf{Q}^{(k)} \mathbf{Q}^{(k)*} - \mathbf{I} \right\| \leq n^{\beta+1} \mathbf{u}. \quad (42)$$

With similar arguments as above, we write

$$\begin{aligned} \mathbf{A}^{(k,2,0)} &= \mathbf{Q}_2 \mathbf{A}^{(k,3,0)} \mathbf{Q}_2^* + \mathbf{E}_2 \\ &= \mathbf{Q}_2 \left(\mathbf{Q}_3 \mathbf{A}^{(k,4,0)} \mathbf{Q}_3^* + \mathbf{E}_3 \right) \mathbf{Q}_2^* + \mathbf{E}_2 \\ &= \mathbf{E}_2 + \sum_{l=3}^{b_k} \left[\left(\prod_{m=3}^l \mathbf{Q}_{m-1} \right) \mathbf{E}_l \left(\prod_{m=3}^l \mathbf{Q}_{m-1} \right)^* \right] + \mathbf{Q}^{(k)} \mathbf{A}^{(k-1,2,0)} \mathbf{Q}^{(k)*}. \end{aligned}$$

Rearranging the terms, we can bound the error norm by

$$\begin{aligned} \left\| \mathbf{A}^{(k,2,0)} - \mathbf{Q} \mathbf{A}^{(k-1,2,0)} \mathbf{Q}^* \right\| &= \left\| \mathbf{E}_2 + \sum_{l=3}^{b_k} \left[\left(\prod_{m=3}^l \mathbf{Q}_{m-1} \right) \mathbf{E}_l \left(\prod_{m=3}^l \mathbf{Q}_{m-1} \right)^* \right] \right\| \\ &\leq \left\| \mathbf{E}_2 \right\| + \sum_{l=3}^{b_k} \left[\left\| \mathbf{E}_l \right\| \left\| \prod_{m=3}^l \mathbf{Q}_{m-1} \right\|^2 \right] \\ &\leq \mathbf{u} n^\beta \cdot C_1 n \left\| \mathbf{A}^{(k,2,0)} \right\| + \sum_{l=3}^{b_k} \left[\mathbf{u} n^\beta C_1 n \left\| \mathbf{A}^{(k,l,0)} \right\| \left(1+n^\beta \mathbf{u} \right)^{l-1} \right] \\ &\leq \mathbf{u} n^\beta \cdot C_1 n \left\| \mathbf{A}^{(k,2,0)} \right\| \left(1 + \sum_{l=3}^{b_k} \left[\left(1+n^\beta \mathbf{u} \right)^{l-1} \left(\frac{1+\mathbf{u}n^\beta C_1 n}{1-n^\beta \mathbf{u}} \right)^{l-1} \right] \right) \end{aligned}$$

Reducing \mathbf{u} further down to $\mathbf{u} \leq \frac{1}{\max\{1, C, C', C_1\} n^{\beta+2}}$ then $(1+n^\beta \mathbf{u})^{l-1} \leq (1+1/n^2)^{b_k} \leq (1+1/n^2)^n \leq c_1$, and

$$\left(\frac{1+\mathbf{u}n^\beta C_1 n}{1-n^\beta \mathbf{u}} \right)^{l-1} \leq \left(\frac{1+1/n}{1-1/n^2} \right)^{b_k} \leq \left(\frac{n}{n-1} \right)^n \leq c_2,$$

where both c_1 and c_2 are constants for all $n \geq 2$. This gives

$$\begin{aligned} \left\| \mathbf{A}^{(k,2,0)} - \mathbf{Q} \mathbf{A}^{(k-1,2,0)} \mathbf{Q}^* \right\| &\leq \mathbf{u} n^\beta \cdot C_1 n \left\| \mathbf{A}^{(k,2,0)} \right\| (1+b_k(1+c_1+c_2)) \\ &\leq \mathbf{u} n^\beta \cdot C_2 n^2 \left\| \mathbf{A}^{(k,2,0)} \right\|. \end{aligned} \quad (43)$$

A bound for the norm of $\mathbf{A}^{(k-1,2,0)}$ is also directly implied

$$\begin{aligned}
\|\mathbf{A}^{(k-1,2,0)}\| &\leq \left\| \left(\mathbf{Q}^{(k)} \right)^{-1} \right\|^2 \left\| \mathbf{Q}^{(k)} \mathbf{A}^{(k-1,2,0)} \mathbf{Q}^{(k)*} \right\| \\
&\leq \left\| \left(\mathbf{Q}^{(k)} \right)^{-1} \right\|^2 \left(\left\| \mathbf{Q}^{(k)} \mathbf{A}^{(k-1,2,0)} \mathbf{Q}^{(k)*} - \mathbf{A}^{(k,2,0)} \right\| + \left\| \mathbf{A}^{(k,2,0)} \right\| \right) \\
&\leq \left(\frac{1}{1 - n^\beta \mathbf{u}} \right)^{b_k-2} \left(1 + C_2 n^\beta \mathbf{u} \right) \left\| \mathbf{A}^{(k,2,0)} \right\| \\
&\leq C_3 \left\| \mathbf{A}^{(k,2,0)} \right\|.
\end{aligned}$$

Each rotation (Lemmas E.1 and E.2) requires $O(n^\omega)$ floating point operations. There are at most $O(b_k^2)$ rotations, which gives a total of $O(n^2 n_k^{\omega-2})$ floating point operations for the rotations. If the final similarity transformation matrix $\widehat{\mathbf{Q}}^{(k)}$ is not required, the algorithm can already terminate. Otherwise, some additional work is required to compute it.

We now describe how to compute the matrix $\widehat{\mathbf{Q}}^{(k)}$ which approximates $\mathbf{Q}^{(k)}$ from the matrices \mathbf{Q}_i in floating point. Each matrix \mathbf{Q}_i is block-diagonal with block size $2n_k$. They can also be seen as block-tridiagonal matrices with block-size n_k . To construct each matrix \mathbf{Q}_i we need to perform $O(b_k n_k^\omega)$ floating point operations. This means that all the matrices \mathbf{Q}_i for $i = 2, \dots, b_k$ can be constructed in $O(b_k^2 n_k^\omega) = O(n^2 n_k^{\omega-2})$ floating point operations.

For any $i \in 2, 4, 6, \dots, b_k - 2$, we can multiply \mathbf{Q}_i with \mathbf{Q}_{i+1} using Corollary E.2. Each such multiplication returns a matrix which is again block-tridiagonal, but with half the number of blocks ($b_k/2$) and with double the block-size ($2n_k$). The cost of the multiplication is $C_{MM} n \cdot n_k^{\omega-1}$ operations. There are $b_k - 2 = n/n_k - 2$ such matrices \mathbf{Q}_i . Therefore, we can perform $(b_k - 2)/2$ block-tridiagonal multiplications to compute all the matrices $\mathbf{Q}_{3:2}, \mathbf{Q}_{5:4}, \dots, \mathbf{Q}_{b_k-1:b_k-2}$, in a total of $\frac{b_k}{2} C_{MM} n \cdot n_k^{\omega-1}$ operations, where each $\mathbf{Q}_{i+1:i}$ is block-tridiagonal with double the block-size $2n_k$. We keep performing the multiplications in a binary-tree structure. At each tree level $l = 2, \dots, \log(b_k) - 1$, we have half the number of block-tridiagonal matrices than in the level $l - 1$, with double the block size. The total complexity becomes

$$\begin{aligned}
T_k &= \sum_{l=1}^{\log(b_k)-1} \overbrace{\left(\frac{b_k}{2^l} \right)}^{\text{number of multiplications at level } l} \cdot \overbrace{C_{MM} n (2^{l-1} n_k)^{\omega-1}}^{\text{cost per multiplication at level } l} \\
&= 2^{1-(\omega)} C_{MM} n \sum_{l=1}^{\log(n)-k-1} \frac{n}{2^{l/2^k}} \left(2^l 2^k \right)^{\omega-1} \\
&= 2^{1-(\omega)} C_{MM} n \sum_{l=k+1}^{\log(n)-1} (2^{\omega-2})^l \\
&= O \left(n^2 (S_\omega(\log(n) - 1) - S_\omega(k)) \right),
\end{aligned}$$

where $S_x(m) := \sum_{l=1}^m (2^{x-2})^l$. In floating point arithmetic, since $\eta > 0$, for every $k = 2, \dots, \log(n) - 2$ the geometric series gives

$$S_\omega(\log(n) - 1) - S_\omega(k) = \frac{(2^{\omega-2})^{\log(n)-1} - (2^{\omega-2})^k}{2^{\omega-2} - 1}.$$

However, if one day it turns out that $\omega = 2$, in exact arithmetic the term becomes $S_2(\log(n)-1) - S_2(k) = \log(n) - 1 - k$.

In terms of the floating point errors, recall first that the matrices in the lowest level of the tree, \mathbf{Q}_i , are approximately orthogonal, i.e. $\mathbf{Q}_i = \mathbf{W}_i + \mathbf{E}_i$ where \mathbf{W}_i is exactly orthogonal and $\|\mathbf{E}_i\| \leq n^\beta \mathbf{u} := \text{err}(0)$. Then we can write $\|\mathbf{Q}_i\| \leq 1 + \text{err}(0)$, and $\text{err}(l)$ will be used to denote the total error at the tree level $l = 1, \dots, \log(b_k)$. Assume that we want to multiply \mathbf{Q}_2 with \mathbf{Q}_3 , to form

$$\widehat{\mathbf{Q}}_1^1 = \text{MM}_{\text{BT}}(\mathbf{Q}_2, \mathbf{Q}_3) = \mathbf{Q}_2 \mathbf{Q}_3 + \mathbf{E}_{\text{MM}_{\text{BT}}} = \mathbf{W}_2 \mathbf{W}_3 + \mathbf{W}_2 \mathbf{E}_3 + \mathbf{E}_2 \mathbf{W}_3 + \mathbf{E}_2 \mathbf{E}_3 + \mathbf{E}_{\text{MM}_{\text{BT}}}.$$

Let $\mathbf{Q}_1^1 = \mathbf{W}_2 \mathbf{W}_3$ be the true product between the unitary matrices \mathbf{W}_2 and \mathbf{W}_3 . We have the following:

$$\begin{aligned}\|\mathbf{E}_{\{2,3\}}\| &\leq \text{err}(0), \\ \|\mathbf{E}_{\text{MMBT}}\| &\leq \mathbf{u} n^\beta \|\mathbf{Q}_2\| \|\mathbf{Q}_3\| \leq \text{err}(0)(1 + \text{err}(0))^2, \\ \|\mathbf{E}_1^1\| &= \|\mathbf{Q}_1^1 - \widehat{\mathbf{Q}}_1^1\| \leq 2\text{err}(0) + \text{err}(0)^2 + \text{err}(0)(1 + \text{err}(0))^2 := \text{err}(1), \\ \|\widehat{\mathbf{Q}}_1^1\| &\leq 1 + \text{err}(1),\end{aligned}$$

Now if we multiply two matrices from level l , e.g. $\widehat{\mathbf{Q}}_1^l$ and $\widehat{\mathbf{Q}}_2^l$, we can write similarly

$$\widehat{\mathbf{Q}}_1^{l+1} = \text{MMBT}(\widehat{\mathbf{Q}}_1^l, \widehat{\mathbf{Q}}_2^l) = \widehat{\mathbf{Q}}_1^l \widehat{\mathbf{Q}}_2^l + \mathbf{E}_{\text{MMBT}} = \mathbf{Q}_1^l \mathbf{Q}_2^l + \mathbf{Q}_1^l \mathbf{E}_2^l + \mathbf{E}_1^l \mathbf{Q}_2^l + \mathbf{E}_1^l \mathbf{E}_2^l + \mathbf{E}_{\text{MMBT}},$$

where $\mathbf{Q}_{\{1,2\}}^l$ are unitary. Denoting $\mathbf{Q}_1^{l+1} = \mathbf{Q}_1^l \mathbf{Q}_2^l$ we have

$$\begin{aligned}\|\mathbf{E}_{\{1,2\}}^l\| &\leq \text{err}(l), \\ \|\mathbf{E}_{\text{MMBT}}\| &\leq \mathbf{u} n^\beta \|\widehat{\mathbf{Q}}_1^l\| \|\widehat{\mathbf{Q}}_2^l\| \leq \text{err}(0)(1 + \text{err}(l))^2, \\ \text{err}(l+1) &= \|\mathbf{E}_1^{l+1}\| = \|\widehat{\mathbf{Q}}_1^{l+1} - \mathbf{Q}_1^{l+1}\| \leq 2\text{err}(l) + \text{err}(l)^2 + \text{err}(0)(1 + \text{err}(l))^2.\end{aligned}$$

A crude solution for the error at level $\log(n)$ is the following. The errors are increasing, i.e. $\text{err}(l+1) \geq \text{err}(l)$. For all l , if $\text{err}(l) \leq 1$, then $\text{err}(l+1) \leq 8\text{err}(l)$. Assume that \mathbf{u} satisfies $\mathbf{u} \leq \frac{1}{n^{\beta+3}}$. Then $\text{err}(0) \leq 1/n^3 \leq 1$, which means that $\text{err}(1) \leq 8\text{err}(0) \leq 8/n^3 \leq 1$. Continuing until level $\log(n-1)$ we obtain that $\text{err}(\log(n)-1) \leq 1/8 \leq 1$. Thus, for every $\mathbf{u} \leq \frac{1}{n^{\beta+3}}$ we can upper bound the error up to level $\log(n)$ with the function

$$\text{err}'(l) = 8\text{err}'(l-1).$$

Then $\text{err}(\log(n)) \leq \text{err}'(\log(n)) \leq 8^{\log(n)} \text{err}(0) = n^3 \text{err}(0) = n^{\beta+3} \mathbf{u}$.

Denoting by $\widehat{\mathbf{Q}}^{(k)}$ the final matrix returned by the floating point multiplications at the last level, and by $\mathbf{Q}^{(k)} = \mathbf{Q}_2 \mathbf{Q}_3 \dots \mathbf{Q}_{b_k-1}$ as above, we obtain

$$\|\widehat{\mathbf{Q}}^{(k)} - \mathbf{Q}^{(k)}\| = \|\mathbf{E}_{\mathbf{Q}}^{(k)}\| \leq \text{err}(\log(b_k)) \leq \text{err}(\log(n)) \leq n^{\beta+3} \mathbf{u},$$

and from above we also already have that $\|\mathbf{Q}^{(k)}\| \leq \sqrt{1 + n^{\beta+1} \mathbf{u}}$. It is straightforward to see that $\widehat{\mathbf{Q}}^{(k)}$ is approximately unitary, since

$$\begin{aligned}\|\widehat{\mathbf{Q}}^{(k)} \widehat{\mathbf{Q}}^{(k)*} - \mathbf{I}\| &= \left\| \left(\mathbf{Q}^{(k)} + \mathbf{E}_{\mathbf{Q}}^{(k)} \right) \left(\mathbf{Q}^{(k)} + \mathbf{E}_{\mathbf{Q}}^{(k)} \right)^* - \mathbf{I} \right\| \\ &\leq \left\| \mathbf{Q}^{(k)} \mathbf{Q}^{(k)*} - \mathbf{I} \right\| + 2 \left\| \mathbf{Q}^{(k)} \mathbf{E}_{\mathbf{Q}}^{(k)} \right\| + \left\| \mathbf{E}_{\mathbf{Q}}^{(k)} \right\|^2 \\ &\leq n^{\beta+1} \mathbf{u} + 2\sqrt{1 + n^{\beta+1} \mathbf{u}} n^{\beta+3} \mathbf{u} + (n^{\beta+3} \mathbf{u})^2 \\ &\leq 6n^{\beta+3} \mathbf{u}.\end{aligned}\tag{44}$$

Denoting by $\mathbf{E} = \widehat{\mathbf{Q}}^{(k)} - \mathbf{Q}^{(k)}$, and combining the aforementioned bound with the bound in 43, we obtain

$$\begin{aligned}\left\| \mathbf{A}^{(k,2,0)} - \widehat{\mathbf{Q}}^{(k)} \mathbf{A}^{(k-1,2,0)} \widehat{\mathbf{Q}}^{(k)*} \right\| &\leq \left\| \mathbf{A}^{(k,2,0)} - \mathbf{Q}^{(k)} \mathbf{A}^{(k-1,2,0)} \mathbf{Q}^{(k)*} \right\| + 2 \left\| \mathbf{E} \mathbf{A}^{(k-1,2,0)} \mathbf{Q}^{(k)*} \right\| + \left\| \mathbf{E} \mathbf{A}^{(k-1,2,0)} \mathbf{E}^* \right\| \\ &\leq \mathbf{u} n^\beta \cdot C_2 n^2 \left\| \mathbf{A}^{(k,2,0)} \right\| + 2 \cdot \mathbf{u} n^3 \zeta \left\| \mathbf{A}^{(k-1,2,0)} \right\| + (\mathbf{u} n^3)^2 \left\| \mathbf{A}^{(k-1,2,0)} \right\| \\ &\leq \left(\mathbf{u} n^{\beta+2} C_2 + 2 \cdot \mathbf{u} n^3 \zeta C_3 + (\mathbf{u} n^3)^2 C_3 \right) \left\| \mathbf{A}^{(k,2,0)} \right\| \\ &\leq \mathbf{u} C_4 n^{\beta+3} \left\| \mathbf{A}^{(k,2,0)} \right\|,\end{aligned}$$

where C_4 is a constant. We conclude that $\widehat{\mathbf{Q}}^{(k)}$ satisfies all the advertised properties.

The requirement for \mathbf{u} is

$$\mathbf{u} \leq \min \left\{ \frac{1}{\max\{1, C, C'\} n^{\beta+1}}, \frac{1}{\max\{1, C, C', C_1\} n^{\beta+2}}, \frac{1}{n^{\beta+3}} \right\},$$

which are all satisfied by setting $\mathbf{u} \leq \frac{1}{C_5 n^{\beta+3}}$, where $C_5 = \max\{1, C, C', C_1\}$. \square

Theorem E.3 (Restatement of Theorem 3.1). *There exists a floating point implementation of the tridiagonal reduction algorithm of [79], which takes as input a Hermitian matrix \mathbf{A} , and returns a tridiagonal matrix $\widetilde{\mathbf{T}}$, and (optionally) an approximately unitary matrix $\widetilde{\mathbf{Q}}$. If the machine precision \mathbf{u} satisfies $\mathbf{u} \leq \epsilon \frac{1}{cn^{\beta+4}}$, where $\epsilon \in (0, 1)$, c is a constant, and β is the same as in Corollary E.1, which translates to $O(\log(n) + \log(1/\epsilon))$ bits of precision, then the following hold:*

$$\|\widetilde{\mathbf{Q}}\widetilde{\mathbf{Q}}^* - \mathbf{I}\| \leq \epsilon, \quad \text{and} \quad \|\mathbf{A} - \widetilde{\mathbf{Q}}\widetilde{\mathbf{T}}\widetilde{\mathbf{Q}}^*\| \leq \epsilon \|\mathbf{A}\|.$$

The algorithm executes at most $O(n^2 S_\omega(\log(n)))$ floating point operations to return only $\widetilde{\mathbf{T}}$, where $S_x(m) = \sum_{l=1}^m (2^{x-2})^l$. If \mathbf{A} is banded with $1 \leq d \leq n$ bands, the floating point operations reduce to $O(n^2 S_\omega(\log(d)))$. If $\widetilde{\mathbf{Q}}$ is also returned, the complexity increases to $O(n^2 C_\omega(\log(n)))$, where $C_x(n) := \sum_{k=2}^{\log(n)-2} (S_x(\log(n)-1) - S_x(k))$. If ω is treated as a constant $\omega \approx 2.371$ the corresponding complexities are $O(n^\omega)$, $O(n^2 d^{\omega-2})$, and $O(n^\omega \log(n))$, respectively.

Proof. We apply the bandwidth halving algorithm recursively. Starting with the original matrix $\mathbf{A} = \mathbf{A}^{(\log(n)-2,2,0)}$, call Algorithm 2 to produce a series of matrices $\widehat{\mathbf{Q}}^{(k)}$, $\mathbf{A}^{(k,2,0)}$ for all $k = \log(n) - 2, \dots, 0$. For $k = 0$, the notation of the final matrix $\mathbf{A}^{(-1,2,0)}$ is symbolic and it denotes that the matrix is tridiagonal.

From Lemma E.3, for all $k = 0, \dots, \log(n) - 2$, it holds that

$$\mathbf{A}^{(k,2,0)} = \widehat{\mathbf{Q}}^{(k)} \mathbf{A}^{(k-1,2,0)} \widehat{\mathbf{Q}}^{(k)*} + \mathbf{E}^{(k)}, \quad \|\mathbf{E}^{(k)}\| \leq C_2 \mathbf{u} n^{\beta+3} \|\mathbf{A}^{(k,2,0)}\|.$$

Combining this with Inequality (42) we have

$$\begin{aligned} \|\mathbf{A}^{(k-1,2,0)}\| &= \left\| \left(\widehat{\mathbf{Q}}^{(k)} \right)^{-1} \left(\mathbf{A}^{(k,2,0)} - \mathbf{E}^{(k)} \right) \left(\widehat{\mathbf{Q}}^{(k)} \right)^{-*} \right\| \\ &\leq \left\| \left(\widehat{\mathbf{Q}}^{(k)} \right)^{-1} \right\|^2 \left(\|\mathbf{A}^{(k,2,0)}\| + \|\mathbf{E}^{(k)}\| \right) \\ &\leq \frac{1+C_2 \mathbf{u} n^{\beta+3}}{1-n^{\beta+1} \mathbf{u}} \|\mathbf{A}^{(k,2,0)}\|. \end{aligned}$$

Note that for the first step $k = \log(n) - 2$, the initial matrix is not block pentadiagonal, but it rather has three block off-diagonals, in which case we need to simply modify the first rotation of Lemma E.3 to include also the additional block. Then

$$\begin{aligned} &\mathbf{A}^{(\log(n)-2,2,0)} \\ &= \widehat{\mathbf{Q}}^{(\log(n)-2)} \mathbf{A}^{(\log(n)-3,2,0)} \widehat{\mathbf{Q}}^{(\log(n)-2)*} + \mathbf{E}^{(\log(n)-2)} \\ &= \widehat{\mathbf{Q}}^{(\log(n)-2)} \left(\widehat{\mathbf{Q}}^{(\log(n)-3)} \mathbf{A}^{(\log(n)-4,2,0)} \widehat{\mathbf{Q}}^{(\log(n)-3)*} + \mathbf{E}^{(\log(n)-3)} \right) \widehat{\mathbf{Q}}^{(\log(n)-2)*} + \mathbf{E}^{(\log(n)-2)} \\ &= \mathbf{E}^{(\log(n)-2)} + \sum_{k=0}^{\log(n)-3} \left[\widehat{\mathbf{Q}}^{(\log(n)-2)} \dots \widehat{\mathbf{Q}}^{(k+1)} \mathbf{E}^{(k)} \widehat{\mathbf{Q}}^{(k+1)*} \dots \widehat{\mathbf{Q}}^{(\log(n)-2)*} \right] + \widehat{\mathbf{Q}} \mathbf{A}^{(-1,2,0)} \widehat{\mathbf{Q}}^*, \end{aligned}$$

Imposing a bound on $\mathbf{u} \leq \frac{\epsilon}{cn^{\beta+4}}$, this gives

$$\|\mathbf{A} - \widetilde{\mathbf{Q}}\widetilde{\mathbf{T}}\widetilde{\mathbf{Q}}^*\| = \|\mathbf{A}^{(\log(n)-2,2,0)} - \widehat{\mathbf{Q}} \mathbf{A}^{(-1,2,0)} \widehat{\mathbf{Q}}^*\|$$

$$\begin{aligned}
&= \left\| \mathbf{E}^{(\log(n)-2)} + \sum_{k=0}^{\log(n)-3} \left[\widehat{\mathbf{Q}}^{(\log(n)-2)} \dots \widehat{\mathbf{Q}}^{(k+1)} \mathbf{E}^{(k)} \widehat{\mathbf{Q}}^{(k+1)*} \dots \widehat{\mathbf{Q}}^{(\log(n)-2)*} \right] \right\| \\
&\leq \left\| \mathbf{E}^{(\log(n)-2)} \right\| + \sum_{k=0}^{\log(n)-3} \left[\left\| \mathbf{E}^{(k)} \right\| \prod_{m=k+1}^{\log(n)-2} \left\| \widehat{\mathbf{Q}}^{(m)} \right\|^2 \right] \\
&\leq C_2 \mathbf{u} n^{\beta+3} \|\mathbf{A}\| + \sum_{k=0}^{\log(n)-2} \left[C_2 \mathbf{u} n^{\beta+3} \left\| \mathbf{A}^{(k,2,0)} \right\| \left(\sqrt{1 + 6n^{\beta+3} \mathbf{u}} \right)^{\log(n)-k-1} \right] \\
&\leq C_2 \mathbf{u} n^{\beta+3} \left(\|\mathbf{A}\| + \sum_{k=0}^{\log(n)-2} \left[\left(\frac{1+C_2 \mathbf{u} n^{\beta+3}}{1-n^{\beta+1} \mathbf{u}} \right)^{\log(n)-k-1} \|\mathbf{A}\| \left(1 + n^{\beta+1} \mathbf{u} \right)^{\log(n)-k-1} \right] \right) \\
&\leq C_2 \mathbf{u} n^{\beta+3} \|\mathbf{A}\| \left(1 + \log(n) \left[\left(\frac{1+C_2 \mathbf{u} n^{\beta+3}}{1-n^{\beta+1} \mathbf{u}} \right)^{\log(n)} \left(1 + n^{\beta+1} \mathbf{u} \right)^{\log(n)} \right] \right) \\
&\leq C_2 \mathbf{u} n^{\beta+3} \|\mathbf{A}\| \left(1 + \log(n) \left[\left(\frac{1+C_2/n}{1-1/n^3} \right)^n \left(1 + 1/n^3 \right)^n \right] \right) \\
&\leq C'_2 \mathbf{u} n^{\beta+4} \|\mathbf{A}\|,
\end{aligned}$$

where we used the assumed bound on \mathbf{u} . $C'_2 \geq C_2$ is a constant.

We finally bound the floating point errors that arise by multiplying the matrices $\widehat{\mathbf{Q}}^{(k)}$ to form $\widetilde{\mathbf{Q}}$, which approximates $\widehat{\mathbf{Q}} = \widehat{\mathbf{Q}}^{(\log(n)-2)} \widehat{\mathbf{Q}}^{(\log(n)-3)} \dots \widehat{\mathbf{Q}}^{(1)} \widehat{\mathbf{Q}}^{(0)}$. Assume the following algorithm:

1. $\widetilde{\mathbf{Q}}^{(0)} = \widehat{\mathbf{Q}}^{(0)}$
2. For $k = 1, \dots, \log(n) - 2$:
3. $\widetilde{\mathbf{Q}}^{(k)} \leftarrow \text{MM}(\widehat{\mathbf{Q}}^{(k)}, \widetilde{\mathbf{Q}}^{(k-1)})$.
4. $\widetilde{\mathbf{Q}} = \widetilde{\mathbf{Q}}^{(\log(n)-2)}$.

Each multiplication introduces an error \mathbf{E}_k^{MM} . Specifically:

$$\widetilde{\mathbf{Q}}^{(k)} = \text{MM}(\widehat{\mathbf{Q}}^{(k)}, \widetilde{\mathbf{Q}}^{(k-1)}) = \widehat{\mathbf{Q}}^{(k)} \widetilde{\mathbf{Q}}^{(k-1)} + \mathbf{E}_k^{\text{MM}},$$

where, from Corollary E.1, $\|\mathbf{E}_k^{\text{MM}}\| \leq \mathbf{u} n^\beta \left\| \widetilde{\mathbf{Q}}^{(k-1)} \right\| \left\| \widehat{\mathbf{Q}}^{(k)} \right\| \leq \mathbf{u} n^\beta \left(\sqrt{1 + 6n^{\beta+3} \mathbf{u}} \right) \left\| \widetilde{\mathbf{Q}}^{(k-1)} \right\|$. Then

$$\begin{aligned}
\left\| \widetilde{\mathbf{Q}}^{(k)} \right\| &\leq \left(1 + \mathbf{u} n^\beta \right) \left(\sqrt{1 + 6n^{\beta+3} \mathbf{u}} \right) \left\| \widetilde{\mathbf{Q}}^{(k-1)} \right\| \\
&\leq \left(1 + \mathbf{u} n^\beta \right)^k \left(\sqrt{1 + 6n^{\beta+3} \mathbf{u}} \right)^k \left\| \widetilde{\mathbf{Q}}^{(0)} \right\| \\
&\leq \left(1 + \mathbf{u} n^\beta \right)^k \left(\sqrt{1 + 6n^{\beta+3} \mathbf{u}} \right)^{k+1}.
\end{aligned}$$

Now take

$$\begin{aligned}
\widetilde{\mathbf{Q}}^{(\log(n)-2)} &= \widehat{\mathbf{Q}}^{(\log(n)-2)} \widetilde{\mathbf{Q}}^{(\log(n)-3)} + \mathbf{E}_{\log(n)-2}^{\text{MM}} \\
&= \widehat{\mathbf{Q}}^{(\log(n)-2)} \left(\widehat{\mathbf{Q}}^{(\log(n)-3)} \widetilde{\mathbf{Q}}^{(\log(n)-4)} + \mathbf{E}_{\log(n)-3}^{\text{MM}} \right) + \mathbf{E}_{\log(n)-2}^{\text{MM}} \\
&= \widehat{\mathbf{Q}}^{(\log(n)-2)} \widehat{\mathbf{Q}}^{(\log(n)-3)} \dots \widehat{\mathbf{Q}}^{(0)} + \mathbf{E}_{\log(n)-2}^{\text{MM}} + \sum_{k=0}^{\log(n)-3} \widehat{\mathbf{Q}}^{(\log(n)-2)} \dots \widehat{\mathbf{Q}}^{(k+1)} \mathbf{E}_k^{\text{MM}},
\end{aligned}$$

which means that

$$\begin{aligned}
& \left\| \widehat{\mathbf{Q}}^{(\log(n)-2)} - \widehat{\mathbf{Q}}^{(\log(n)-2)} \dots \widehat{\mathbf{Q}}^{(1)} \widehat{\mathbf{Q}}^{(0)} \right\| \\
& \leq \left\| \mathbf{E}_{\log(n)-2}^{\text{MM}} \right\| + \sum_{k=1}^{\log(n)-3} \left\| \mathbf{E}_k^{\text{MM}} \right\| \prod_{l=k+1}^{\log(n)-2} \left\| \widehat{\mathbf{Q}}^{(l)} \right\| \\
& \leq \sum_{k=1}^{\log(n)-2} \left\| \mathbf{E}_k^{\text{MM}} \right\| \left(\sqrt{1 + 6n^{\beta+3}\mathbf{u}} \right)^{\log(n)-2-k} \\
& \leq \sum_{k=1}^{\log(n)-2} \mathbf{u} n^{\beta} \left(\sqrt{1 + 6n^{\beta+3}\mathbf{u}} \right) \left\| \widehat{\mathbf{Q}}^{(k-1)} \right\| \left(\sqrt{1 + 6n^{\beta+3}\mathbf{u}} \right)^{\log(n)-2-k} \\
& \leq \mathbf{u} n^{\beta} \left(\sqrt{1 + 6n^{\beta+3}\mathbf{u}} \right)^{\log(n)-2} \sum_{k=1}^{\log(n)-2} \left(1 + \mathbf{u} n^{\beta} \right)^{k-1} \left(\sqrt{1 + 6n^{\beta+3}\mathbf{u}} \right)^k \\
& \leq \mathbf{u} n^{\beta} \left(\sqrt{1 + 6n^{\beta+3}\mathbf{u}} \right)^{\log(n)-2} (\log(n) - 2) \left(1 + \mathbf{u} n^{\beta} \right)^{\log(n)-3} \left(\sqrt{1 + 6n^{\beta+3}\mathbf{u}} \right)^{\log(n)-2} \\
& \leq \mathbf{u} n^{\beta+1} \left(1 + 6n^{\beta+3}\mathbf{u} \right)^{\log(n)-2} \left(1 + \mathbf{u} n^{\beta} \right)^{\log(n)-3} \\
& \leq \mathbf{u} n^{\beta+1} (1 + 1/n)^n \\
& \leq C_Q \mathbf{u} n^{\beta+1},
\end{aligned}$$

where C_Q is a small constant (the bounds are quite loose, but it does not affect the asymptotic complexity analysis as long as they stay polynomial in n).

Next, we recall that the matrix $\widehat{\mathbf{Q}}$ is approximately orthogonal. From Lemma E.3, for each $k = 0, \dots, \log(n) - 2$:

$$\left\| \widehat{\mathbf{Q}}^{(k)} \dots \widehat{\mathbf{Q}}^{(1)} \widehat{\mathbf{Q}}^{(0)} \right\| \leq \left(\sqrt{1 + n^{\beta+1}\mathbf{u}} \right)^{k+1}, \quad \text{and} \quad \left\| \left(\widehat{\mathbf{Q}}^{(k)} \dots \widehat{\mathbf{Q}}^{(1)} \widehat{\mathbf{Q}}^{(0)} \right)^{-1} \right\| \leq \left(\frac{1}{\sqrt{1 - n^{\beta+1}\mathbf{u}}} \right)^{k+1}.$$

Since $n^{\beta+1}\mathbf{u} \leq 1/n^2 \ll 1$, using again Bernoulli's inequalities it follows that

$$\begin{aligned}
\left\| \widehat{\mathbf{Q}} \right\|^2 & \leq (1 + n^{\beta+1}\mathbf{u})^{\log(n)-1} \leq (1 + n^{\beta+1}\mathbf{u})^n \leq 1 + n^{\beta+2}\mathbf{u}, \\
\left\| \widehat{\mathbf{Q}}^{-1} \right\|^{-2} & \geq (1 - n^{\beta+1}\mathbf{u})^{\log(n)-1} \geq 1 - (\log(n) - 1)n^{\beta+1}\mathbf{u} \geq 1 - n^{\beta+2}\mathbf{u},
\end{aligned}$$

and since all the (squared) singular values of $\widehat{\mathbf{Q}}$ are inside $[1 - n^{\beta+2}\mathbf{u}, 1 + n^{\beta+2}\mathbf{u}]$ then

$$\left\| \widehat{\mathbf{Q}} \widehat{\mathbf{Q}}^* - \mathbf{I} \right\| \leq n^{\beta+2}\mathbf{u}. \tag{45}$$

We can now bound the advertised error. First, if we denote $\mathbf{E}_A = \mathbf{A} - \widehat{\mathbf{Q}} \widetilde{\mathbf{T}} \widehat{\mathbf{Q}}^*$ then

$$\begin{aligned}
\left\| \widetilde{\mathbf{T}} \right\| & = \left\| \widehat{\mathbf{Q}}^{-1} (\mathbf{A} - \mathbf{E}_A) \widehat{\mathbf{Q}}^{-*} \right\| \\
& \leq \left\| \widehat{\mathbf{Q}}^{-1} \right\|^2 (\|\mathbf{A}\| + \|\mathbf{E}_A\|) \\
& \leq \frac{(1 + C'_2 \mathbf{u} n^{\beta+4})}{1 - n^{\beta+2}\mathbf{u}} \|\mathbf{A}\|
\end{aligned}$$

Denoting by $\mathbf{E}_{\hat{\mathbf{Q}}} = \tilde{\mathbf{Q}} - \hat{\mathbf{Q}}$, we have that

$$\begin{aligned}
\|\mathbf{A} - \tilde{\mathbf{Q}}\tilde{\mathbf{T}}\tilde{\mathbf{Q}}^*\| &\leq \|\mathbf{A} - \tilde{\mathbf{Q}}\tilde{\mathbf{T}}\hat{\mathbf{Q}}\| + 2\|\mathbf{E}_{\hat{\mathbf{Q}}}\|\|\hat{\mathbf{Q}}\|\|\tilde{\mathbf{T}}\| + \|\mathbf{E}_{\hat{\mathbf{Q}}}\|^2\|\tilde{\mathbf{T}}\| \\
&\leq C'_2\mathbf{u}n^{\beta+4}\|\mathbf{A}\| + \left(2C_Q\mathbf{u}n^{\beta+1}\sqrt{1+n^{\beta+2}\mathbf{u}} + (C_Q\mathbf{u}n^{\beta+1})^2\right) \frac{(1+C'_2\mathbf{u}n^{\beta+4})}{1-n^{\beta+2}\mathbf{u}}\|\mathbf{A}\| \\
&\leq C_{\text{TRID}}\mathbf{u}n^{\beta+4}\|\mathbf{A}\|,
\end{aligned} \tag{46}$$

where C_{TRID} is a constant. Note also that

$$\begin{aligned}
\|\tilde{\mathbf{Q}}\tilde{\mathbf{Q}}^* - \mathbf{I}\| &\leq \|\tilde{\mathbf{Q}}\tilde{\mathbf{Q}}^* - \hat{\mathbf{Q}}\hat{\mathbf{Q}}^*\| + \|\hat{\mathbf{Q}}\hat{\mathbf{Q}}^* - \mathbf{I}\| \\
&\leq 2\|\mathbf{E}_{\hat{\mathbf{Q}}}\|\|\hat{\mathbf{Q}}\| + \|\mathbf{E}_{\hat{\mathbf{Q}}}\|^2 + \mathbf{u}n^{\beta+4} \\
&\leq 2C_Q\mathbf{u}n^{\beta+1}\sqrt{1+n^{\beta+2}\mathbf{u}} + (C_Q\mathbf{u}n^{\beta+1})^2 + \mathbf{u}n^{\beta+4} \\
&\leq C'_Q\mathbf{u}n^{\beta+4}.
\end{aligned}$$

It remains to bound the complexity. Depending whether the matrix $\tilde{\mathbf{Q}}$ is returned or not, we have the following two cases based on Lemma E.3.

- (i) If the matrix $\tilde{\mathbf{Q}}$ must be returned, $O(n^2(S_\omega(\log(n)-1) - S_\omega(k)))$ floating point operations are required during each iteration $k = 2, \dots, \log(n)-2$, which gives a total of

$$O\left(n^2 \sum_{k=2}^{\log(n)-2} (S_\omega(\log(n)-1) - S_\omega(k))\right) = O(n^2 C_\omega(n)).$$

floating point operations, where we used the notation $C_x(n) := \sum_{k=2}^{\log(n)-2} (S_x(\log(n)-1) - S_x(k))$.

- (ii) If we ignore $\tilde{\mathbf{Q}}$ and only return $\tilde{\mathbf{T}}$, only $O(n^2(2^{\omega-2})^k)$ floating point operations are executed per iteration k . In this case the total complexity is

$$O(n^2) \sum_{k=2}^{\log(n)-2} (2^{\omega-2})^k = O(n^2 S_\omega(\log(n)))$$

floating point operations.

If one day it turns out that $\omega = 2$, the corresponding complexities in the limit $\eta \rightarrow 0$ become as follows.

- (i) If we return both $\tilde{\mathbf{T}}$ and $\tilde{\mathbf{Q}}$, the total complexity is

$$O\left(n^2 \sum_{k=2}^{\log(n)-2} S_2(\log(n)-1) - S_2(k)\right) = O(n^2 \log^2(n)).$$

- (ii) On the other hand, if only $\tilde{\mathbf{T}}$ is returned, then the complexity becomes

$$O(n^2 S_2(\log(n))) = O(n^2 \log(n)).$$

Moreover, if \mathbf{A} is banded with bandwidth $1 \leq b \leq n-1$ (number of off-diagonals), then we start the bandwidth-halving recursion from $k = \min\{j \in \mathbb{N} | 2^j > b\}$. This gives a complexity of $O(n^2 S_2(\log(n))) = O(n^2 \log(d))$.

□

We have the following direct Corollary.

Corollary E.3. *The eigenvalues of the matrix $\tilde{\mathbf{T}}$ returned by the algorithm of Theorem 3.1 satisfy*

$$\left| \lambda_i(\tilde{\mathbf{T}}) - \lambda_i(\mathbf{A}) \right| \leq \epsilon \|\mathbf{A}\|.$$

Proof. Since $\tilde{\mathbf{Q}}$ is approximately unitary, it can be written as $\tilde{\mathbf{Q}} = \mathbf{Q} + \mathbf{E}_Q$, where \mathbf{Q} is unitary (exactly) and $\|\mathbf{E}_Q\| \leq C'_Q \mathbf{u} n^{\beta+4}$. Then we can finally bound the error

$$\begin{aligned} \left\| \mathbf{A} - \mathbf{Q} \tilde{\mathbf{T}} \mathbf{Q}^* \right\| &\leq \left\| \mathbf{A} - \tilde{\mathbf{Q}} \tilde{\mathbf{T}} \tilde{\mathbf{Q}}^* \right\| + 2 \|\mathbf{E}_Q\| \left\| \tilde{\mathbf{T}} \right\| + \|\mathbf{E}_Q\|^2 \left\| \tilde{\mathbf{T}} \right\| \\ &\leq \mathbf{u} n^{\beta+4} \|\mathbf{A}\| \left(C_{\text{TRID}} + C'_Q \left(2 + C'_Q \mathbf{u} n^{\beta+4} \right) \frac{(1 + C'_2 \mathbf{u} n^{\beta+4})}{1 - n^{\beta+2} \mathbf{u}} \right) \\ &\leq C'_{\text{TRID}} \mathbf{u} n^{\beta+4} \|\mathbf{A}\| \\ &\leq \epsilon \|\mathbf{A}\|, \end{aligned}$$

where the last holds by assuming that $\mathbf{u} = \Theta(\epsilon/n^{\beta+4})$, like in Theorem 3.1. Since \mathbf{Q} is unitary, it means that the matrix $\mathbf{Q} \tilde{\mathbf{T}} \mathbf{Q}^*$ is similar to $\tilde{\mathbf{T}}$, and the eigenvalues of $\mathbf{Q} \tilde{\mathbf{T}} \mathbf{Q}^*$ satisfy $\left| \lambda_i(\mathbf{Q} \tilde{\mathbf{T}} \mathbf{Q}^*) - \lambda_i(\mathbf{A}) \right| \leq \epsilon \|\mathbf{A}\|$ from Weyl's inequality. \square