Solving Sparse Linear Systems Faster than Matrix Multiplication

Richard Peng
Georgia Tech
rpeng@cc.gatech.edu

Santosh Vempala
Georgia Tech
vempala@gatech.edu

January 8, 2021

Abstract

Can linear systems be solved faster than matrix multiplication? While there has been remarkable progress for the special cases of graph structured linear systems, in the general setting, the bit complexity of solving an $n \times n$ linear system $Ax = b$ is $\widetilde{O}(n^{\omega})$, where $\omega < 2.372864$ is the matrix multiplication exponent. Improving on this has been an open problem even for sparse linear systems with poly(n) condition number.

In this paper, we present an algorithm that solves linear systems in sparse matrices asymptotically faster than matrix multiplication for any $\omega > 2$. This speedup holds for any input matrix A with $o(n^{\omega-1}/\log(\kappa(A)))$ non-zeros, where $\kappa(A)$ is the condition number of A. For poly(n)-conditioned matrices with $\widetilde{O}(n)$ nonzeros, and the current value of ω, the bit complexity of our algorithm to solve to within any $1/\text{poly}(n)$ error is $O(n^{2.331645})$.

Our algorithm can be viewed as an efficient, randomized implementation of the block Krylov method via recursive low displacement rank factorizations. It is inspired by the algorithm of [Eberly et al. ISSAC '06 '07] for inverting matrices over finite fields. In our analysis of numerical stability, we develop matrix anti-concentration techniques to bound the smallest eigenvalue and the smallest gap in eigenvalues of semi-random matrices.

Contents

1 Introduction

Solving a linear system $Ax = b$ is a basic algorithmic problem with direct applications to scientific computing, engineering, and physics, and is at the core of algorithms for many other problems, including optimization [Ye11], data science [BHK20], and computational geometry [EH10]. It has enjoyed an array of elegant approaches, from Cramer's rule and Gaussian elimination to numerically stable iterative methods to more modern randomized variants based on random sampling [ST11, KMP12] and sketching [DMM08, Woo14]. Despite much recent progress on faster solvers for graph-structured linear systems [Vai89, Gre96, ST14, KMP12, Kyn17], progress on the general case has been elusive.

Most of the work in obtaining better running time bounds for linear systems solvers has focused on efficiently computing the inverse of A, or some factorization of it. Such operations are in turn closely related to the cost of matrix multiplication. Matrix inversion can be reduced to matrix multiplication via divide-and-conquer, and this reduction was shown to be stable when the word size for representing numbers[1] is increased by a factor of $O(\log n)$ [DDHK07]. The current best runtime of $O(n^\omega)$ with $\omega < 2.372864$ [LG14] follows a long line of work on faster matrix multiplication algorithms [Str69, Pan84, CW87, Wil12, LG14] and is also the current best running time for solving $Ax = b$: when the input matrix/vector are integers, matrix multiplication based algorithms can obtain the exact rational value solution using $O(n^\omega)$ word operations [Dix82, Sto05].

Methods for matrix inversion or factorization are often referred to as direct methods in the linear systems literature [DRSL16]. This is in contrast to iterative methods, which gradually converge to the solution. Iterative methods have little space overhead, and therefore are widely used for solving large, sparse, linear systems that arise in scientific computing. Another reason for their popularity is that they are naturally suited to producing approximate solutions of desired accuracy in floating point arithmetic, the de facto method for representing real numbers. Perhaps the most famous iterative method is the Conjugate Gradient (CG) / Lanczos algorithm [HS52, Lan50]. It was introduced as an $O(n \cdot nnz)$ time algorithm under exact arithmetic, where nnz is the number of non-zeros in the matrix. However, this bound only holds under the Real RAM model where the words have with unbounded precision [PS85, BSS$^+$89]. When taking bit sizes into account, it incurs an additional factor of n. Despite much progress in iterative techniques in the intervening decades, obtaining analogous gains over matrix multiplication in the presence of round-off errors has remained an open question.

The convergence and stability of iterative methods typically depends on some *condition number* of the input. When all intermediate steps are carried out to precision close to the

[1]We will be measuring bit-complexity under fixed-point arithmetic. Here the machine word size is on the order of the maximum number of digits of precision in A, and the total cost is measured by the number of word operations. The need to account for bit-complexity of the numbers naturally led to the notion of condition number [Tur48, Blu04]. The logarithm of the condition number measures the additional number of words needed to store A^{-1} (and thus $A^{-1}b$) compared to A. In particular, matrices with *poly*(n) condition number can be stored with a constant factor overhead in precision, and are numerically stable under standard floating point number representations.

condition number of A, the running time bounds of the conjugate gradient algorithm, as well as other currently known iterative methods, depend polynomially on the condition number of the input matrix A. Formally, the condition number of a symmetric matrix A, $\kappa(A)$, is the ratio between the maximum and minimum eigenvalues of A. Here the best known rate of convergence when all intermediate operations are restricted to bit-complexity $O(\log(\kappa(A)))$ is to an error of ϵ in $O(\sqrt{\kappa(A)}\log(1/\epsilon))$ iterations. This is known to be tight if one restricts to matrix-vector multiplications in the intermediate steps [SV13, MMS18]. This means for moderately conditioned (e.g. with $\kappa = poly(n)$), sparse, systems, the best runtime bounds are still via direct methods, which are stable when $O(\log(1/\kappa))$ words of precision are maintained in intermediate steps [DDHK07].

Many of the algorithms used in scientific computing for solving linear systems involving large, space, matrices are based on combining direct and iterative methods: we will briefly discuss this perspectives in Section 1.3. From the asymptotic complexity perspective, the practical successes of many such methods naturally leads to the question of whether one can provably do better than the $O(\min\{n^{\omega}, nnz\cdot\sqrt{\kappa(A)}\})$ time corresponding to the faster of direct or iterative methods. Somewhat surprisingly, despite the central role of this question in scientific computing and numerical analysis, as well as extensive studies of linear systems solvers, progress on this question has been elusive. The continued lack of progress on this question has led to its use as a hardness assumption for showing conditional lower bounds for numerical primitives such as linear elasticity problems [KZ17] and positive linear programs [KWZ20]. One formalization of such hardness is the *Sparse Linear Equation Time Hypothesis* (SLTH) from [KWZ20]: SLTH_k^{γ} denotes the assumption that a sparse linear system with $\kappa \leqslant nnz(A)^k$ cannot be solved in time faster than $nnz(A)^{\gamma}$ to within relative error $\epsilon = n^{-10k}$. Here improving over the smaller running time of both direct and iterative methods can be succinctly encapsulated as refuting $\mathrm{SLTH}_k^{\min\{1+k/2,\omega\}}$. [2]

In this paper, we provide a faster algorithm for solving sparse linear systems. Our formal result is the following (we use the form defined in [KWZ20] [Linear Equation Approximation Problem, LEA]).

Theorem 1.1. *Given a matrix A with max dimension n, $nnz(A)$ non-zeros (whose values fit into a single word), along with a parameter $\kappa(A)$ such that $\kappa(A) \geqslant \sigma_{\max}(A)/\sigma_{\min}(A)$, along with a vector b and error requirement ϵ, we can compute, under fixed point arithmetic, in time*

$$O\left(\max\left\{nnz(A)^{\frac{\omega-2}{\omega-1}}n^2, n^{\frac{5\omega-4}{\omega+1}}\right\}\log^2(\kappa/\epsilon)\right)$$

a vector x such that

$$\|Ax - \Pi_A b\|_2^2 \leqslant \epsilon\,\|\Pi_A b\|_2^2\,,$$

where c is a fixed constant and Π_A is the projection operator onto the column space of A.

Note that $\|\Pi_A b\|_2 = \|A^T b\|_{(A^TA)^{-1}}$, and when A is square and full rank, it is just $\|b\|_2$.

[2]The hardness results in [KWZ20] were based on $\mathrm{SLTH}_{1.5}^{1.99}$ under the Real RAM model in part due to the uncertain status of conjugate gradient in different models of computation.

The cross-over point for the two bounds is at $nnz(A) = n^{\frac{3(\omega-1)}{\omega+1}}$. In particular, for the sparse case with $nnz(A) = O(n)$, and the current best $\omega \leqslant 2.372864$ [LG14], we get an exponent of

$$\max\left\{2 + \frac{\omega - 2}{\omega - 1}, \frac{5\omega - 4}{\omega + 1}\right\} < \max\{2.271595, 2.331645\} = 2.331645.$$

As $n \leqslant nnz$, this also translates to a running time of $O(nnz^{\frac{5\omega-4}{\omega+1}})$, which as $\frac{5\omega-4}{\omega+1} = \omega - \frac{(\omega-2)^2}{\omega+1}$, refutes SLTH_k^ω for constant values of k and any value of $\omega > 2$.

We can parameterize the asymptotic gains over matrix multiplication for moderately sparse instances. Here we use the $\widetilde{O}(\cdot)$ notation to hide lower-order terms, specifically $\widetilde{O}(f(n))$ denotes $O(f(n) \cdot \log^c(f(n)))$ for some absolute constant c.

Corollary 1.2. *For any matrix A with dimension at most n, $O(n^{\omega-1-\theta})$ non-zeros, and condition number $n^{O(1)}$, a linear system in A can be solved to accuracy $n^{-O(1)}$ in time $\widetilde{O}(\max\{n^{\frac{5\omega-4}{\omega+1}}, n^{\omega-\frac{\theta(\omega-2)}{\omega-1}}\})$.*

Here the cross-over point happens at $\theta = \frac{(\omega-1)(\omega-2)}{\omega+1}$. Also, because $\frac{5\omega-4}{\omega+1} = \omega - \frac{(\omega-2)^2}{\omega+1}$, we can also infer that for any $0 < \theta \leqslant \omega - 2$ and any $\omega > 2$, the runtime is $o(n^\omega)$, or asymptotically faster than matrix multiplication.

1.1 Idea

At a high level, our algorithm follows the block Krylov space method (see e.g. Chapter 6.12 of Saad [Saa03]). This method is a multi-vector extension of the conjugate gradient / Lanczos method, which in the single-vector setting is known to be problematic under round-off errors both in theory [MMS18] and in practice [GO89]. Our algorithm starts with a set of s initial vectors, $B \in \Re^{n \times s}$, and forms a column space by multiplying these vectors by A repeatedly, m times. Formally, the block Krylov space matrix is

$$K = \left[\ B \mid AB \mid A^2B \mid \ldots \mid A^{m-1}B\ \right].$$

The core idea of Krylov space methods is to efficiently orthogonalize this column space. For this space to be spanning, block Krylov space methods typically choose s and m so that $sm = n$.

The conjugate gradient algorithm can be viewed as an efficient implementation of the case $s = 1$, $m = n$, and B is set to b, the RHS of the input linear system. The block case with larger values of s was studied by Eberly, Giesbrecht, Giorgi, Storjohann, and Villard [EGG+06, EGG+07] over finite fields, and they gave an $O(n^{2.28})$ time algorithm for computing the inverse of a sparse matrix over a finite field.

Our algorithm also leverages the top-level insight of the Eberly et al. results: the Gram matrix of the Krylov space matrix (which can be used inter-changeably for solving linear systems) is a block Hankel matrix. That is, if we view the Gram matrix $(AK)^T(AK)$ as an

m-by-m matrix containing s-by-s sized blocks, then all the blocks along each anti-diagonal are the same:

$$(AK)^T(AK) = \left[\begin{array}{c|c|cccc} B^TA^2B & B^TA^3B & B^TA^4B & \dots & B^TA^{m+1}B \\ \hline B^TA^3B & B^TA^4B & B^TA^5B & \dots & B^TA^{m+2}B \\ \hline B^TA^4B & B^TA^5B & B^TA^6B & \dots & B^TA^{m+3}B \\ \dots & \dots & \dots & \dots & \dots \\ B^TA^{m+1}B & B^TA^{m+2}B & B^TA^{m+3}B & \dots & B^TA^{2m}B \end{array}\right]$$

Formally, the s-by-s inner product matrix formed from A^iB and A^jB is $B^TA^{i+j}B$, and only depends on $i+j$. So instead of m^2 blocks each of size $s \times s$, we are able to represent a n-by-n matrix with about m blocks.

Operations involving these m blocks of the Hankel matrix can be handled using $\widetilde{O}(m)$ block operations. This is perhaps easiest seen for computing matrix-vector products using K. If we use $\{i\}$ to denote the ith block of the Hankel matrix, that is

$$H_{\{i,j\}} = M(i+j)$$

for a sequence of matrices M, we get that the i^{th} block of the product Hx can be written in block-form as

$$(Hx)_{\{i\}} = \sum_j H_{\{i,j\}} x_{\{j\}} = \sum_j M(i+j)\, x_{\{j\}}.$$

Note this is precisely the convolution of (a sub-interval) of M and x, with shifts indicated by i. Therefore, in the forward matrix-vector multiplication direction, a speedup by a factor of about m is possible with fast convolution algorithms. The performance gains of the Eberly et al. algorithms [EGG+06, EGG+07] can be viewed as of similar nature, albeit in the more difficult direction of solving linear systems. Specifically, they utilize algorithms for the Padé problem of computing a polynomial from the result of its convolution [XB90, BL94]. Over finite fields, or under exact arithmetic, such algorithms for matrix Padé problems take $O(m \log m)$ block operations [BL94], for a total of $\widetilde{O}(s^\omega m)$ operations..

The overall time complexity follows from two opposing goals:

1. Quickly generate the Krylov space: repeated multiplication by A allows us to generate A^iB using $O(ms \cdot nnz) = O(n \cdot nnz)$ arithmetic operations. Choosing a sparse B then allows us to compute B^TA^iB in $O(n \cdot s)$ arithmetic operations, for a total overhead of $O(n^2) = O(n \cdot nnz)$.

2. Quickly invert the Hankel matrix. Each operation on an s-by-s block takes $O(s^\omega)$ time. Under the optimistic assumption of $\widetilde{O}(m)$ block operations, the total is $\widetilde{O}(m \cdot s^\omega)$.

Under these assumptions, and the requirement of $n \approx ms$, the total cost becomes about $O(n \cdot nnz + m \cdot s^\omega)$, which is at most $O(n \cdot nnz)$ as long as $m > n^{\frac{\omega-2}{\omega-1}}$. However, this runtime complexity is over finite fields, where numerical stability is not an issue, instead of over reals under round-off errors, where one must contend with numerical errors without blowing up the bit complexity. This is a formidable challenge; indeed, with exact arithmetic, the CG method takes time $O(n \cdot nnz)$, but this is misleading since the computation is effective only the word sizes increase by a factor of n (to about $n \log \kappa$ words), which leads to an overall complexity of $O(n^2 \cdot nnz \cdot \log \kappa)$.

1.2 Our Contributions

Our algorithm can be viewed as the numerical generalization of the algorithms from [EGG+06, EGG+07]. We work with real numbers of bounded precision, instead of entries over a finite field. The core of our approach can be summarized as:

> The block Krylov space method together with fast Hankel solvers can be made numerically stable using $\widetilde{O}(m \log(\kappa))$ words of precision.

Doing so, on the other hand, requires developing tools for two topics that have been extensively studied in mathematics, but separately.

1. Obtain low numerical cost solvers for block Hankel/Toeplitz matrices. Many of the prior algorithms rely on algebraic identities that do not generalize to the block setting, and are often (experimentally) numerically unstable [GTVDV96, Gra06].

2. Develop matrix anti-concentration bounds for analyzing the word lengths of inverses of random Krylov spaces. Such bounds upper bound the probability of random matrices being in some set of small measure, which in our case is the set of nearly singular matrices. Previously, they were known assuming the matrix entries are independent [SST06, TV10], while Krylov spaces have correlated columns.

Furthermore, due to the shortcomings of the matrix anti-concentration bounds, we modify the solver algorithm so that it uses a more limited version of the block-Krylov space that fall under the cases that could be analyzed.

Before we describe the difficulties and new tools needed, we first provide some intuition on why a factor m increase in word lengths may be the right answer by upper-bounding the magnitudes of entries in a m-step Krylov space. The maximum magnitude of $A^m b$ is bounded by the max magnitude of A to the power of m, times a factor corresponding to the number of summands in the matrix product:

$$\left\|A^m b\right\|_\infty \leqslant \left(n \left\|\left|A\right\|\right|_\infty\right)^m \left\|b\right\|_\infty .$$

So the largest numbers in K (as well as AK) can be bounded by $(n\kappa)^{O(m)}$, or $O(m \log \kappa)$ words in front of the decimal point under the assumption of $\kappa > n$.

Should such a bound of $O(m \log \kappa)$ hold for all numbers that arise, including the matrix inversions, and the matrix B is sparse with $O(n)$ entries, the cost of computing the block-Krylov matrices becomes $O(m \log \kappa \cdot ms \cdot nnz)$, while the cost of the matrix inversion portion encounters an overhead of $O(m \log \kappa)$, for a total of $\widetilde{O}(m^2 s^\omega \log \kappa)$. In the sparse case of $nnz = O(n)$, and $n \approx ms$, this becomes:

$$O\left(n^2 m \log \kappa + m^2 s^\omega \log \kappa\right) = O\left(n^2 m \log \kappa + \frac{n^\omega}{m^{\omega-2}} \log \kappa\right). \tag{1}$$

Due to the gap between n^2 and n^ω, setting m appropriately gives improvement over n^ω when $\log \kappa < n^{o(1)}$.

However, the magnitude of an entry in the inverse depends on the smallest magnitude, or in the matrix case, its minimum singular value. Bounding and propagating the min singular value, which intuitively corresponds to how close a matrix is to being degenerate, represents our main challenge. In exact/finite fields settings, non-degeneracies are certified via the Schwartz-Zippel Lemma about polynomial roots. The numerical analog of this is more difficult: the Krylov space matrix K is asymmetric, even for a symmetric matrix A. It is much easier for an asymmetric matrix with correlated entries to be close to singular.

Consider for example a two-banded, two-block matrix with all diagonal entries set to the same random variable α (see Figure 1):

$$A_{ij} = \begin{cases} 1 & \text{if } i = j \text{ and } j \leqslant n/2, \\ \alpha & \text{if } i = j+1 \text{ and } j \leqslant n/2, \\ \alpha & \text{if } i = j+1 \text{ and } n/2 < j, \\ 2 & \text{if } i = j+1 \text{ and } n/2 < j, \\ 0 & \text{otherwise.} \end{cases}$$

In the exact case, this matrix is full rank unless $\alpha = 0$, even over finite fields. On the other hand, its minimum singular value is close to 0 for all values of α because:

Observation 1.3. *The minimum singular value of a matrix with* 1*s on the diagonal,* α *on the entries immediately below the diagonal, and* 0 *everywhere else is at most* $|\alpha|^{-(n-1)}$*, due to the test vector* $[1; -\alpha; \alpha^2; \ldots; (-\alpha)^{n-1}]$.

Specifically, in the top-left block, as long as $|\alpha| > 3/2$, the top left block has minimum singular value at most $(2/3)^{n-1}$. On the other hand, rescaling the bottom-right block by $1/\alpha$ to get 1s on the diagonal gives $2/\alpha$ on the off-diagonal. So as long as $|\alpha| < 3/2$, this value is at least $4/3$, which in turn implies a minimum singular value of at most $(3/4)^{n-1}$ in the bottom right block. This means no matter what value α is set to, this matrix will always have a singular value that's exponentially close to 0. Furthermore, the Gram matrix of this matrix also gives such a counter example to symmetric matrices with (non-linearly) correlated entries. Previous works on analyzing condition numbers of asymmetric matrices also encounter similar difficulties: a more detailed discussion of it can be found in Section 7 of Sankar et al. [SST06].

In order to bound the bit complexity of all intermediate steps of the block Krylov algorithm by $\widetilde{O}(m) \cdot \log \kappa$, we devise a more numerically stable algorithm for solving block

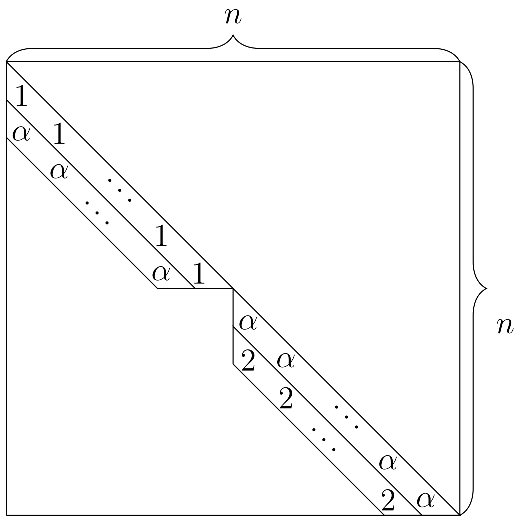

Figure 1: The difference between matrix anti-concentration over finite fields and reals: a matrix that is full rank for all $\alpha \neq 0$, but is always ill conditioned.

Hankel matrices, as well as provide a new perturbation scheme to quickly generate a well-conditioned block Krylov space. Central to both of our key components is the close connection between condition number and bit complexity bounds.

First, we give a more numerically stable solver for block Hankel/Toeplitz matrices. Fast solvers for Hankel (and closely related Toeplitz) matrices have been extensively studied in numerical analysis, with several recent developments on more stable algorithms [XXG12]. However, the notion of numerical stability studied in these algorithms is the more practical variant where the number of bits of precision is fixed. As a result, the asymptotic behavior of the stable algorithm from [XXG12] is quadratic in the number of digits in the condition number, which in our case would translate to a prohibitive cost of $\widetilde{O}(m^2)$ (i.e., the overall cost would be higher than n^ω).

Instead, we combine developments in recursive block Gaussian elimination [DDHK07, KLP+16, CKK+18] with the *low displacement rank* representation of Hankel/Toeplitz matrices [KKM79, BA80]. Such representations allow us to implicitly express both the Hankel matrix and its inverse by displaced versions of rank $2s$ matrices. This means the intermediate sizes of instances arising from recursion is $O(s)$ times the dimension, for a total size of $O(n \log n)$, giving a total of $\widetilde{O}(ns^{\omega-1})$ arithmetic operations involving words of size $\widetilde{O}(m)$. We provide a rigorous analysis of the accumulation of round-off errors similar to the analysis of recursive matrix multiplication based matrix inversion from [DDHK07].

Motivated by this close connection with the condition number of Hankel matrices, we then try to initialize with Krylov spaces of low condition number. Here we show that a sufficiently small perturbation suffices for producing a well conditioned overall matrix. In fact, the first step of our proof, that a small sparse random perturbation to A guarantees good separations between its eigenvalues is a direct combination of bounds on eigenvalue separation of random Gaussians [NTV17] as well as min eigenvalue of random sparse matrices [LV18]. This separation then ensures that the powers of A, $A^1, A^2, \ldots A^m$, are sufficiently distinguishable from each other. Such considerations also come up in the smoothed analysis of numerical algorithms [SST06].

The randomness of the Krylov matrix induced by the initial set of random vectors B is more difficult to analyze: each column of B affects m columns of the overall Krylov space matrix. In contrast, all existing analyses of lower bounds of singular values of possibly asymmetric random matrices [SST06, TV10] rely on the randomness in the columns of matrices being independent. The dependence between columns necessitates analyzing singular values of random linear combinations of matrices, which we handle by adapting ϵ-net based proofs of anti-concentration bounds. Here we encounter an additional challenge in bounding the minimum singular value of the block Krylov matrix. We resolve this issue algorithmically: instead of picking a Krylov space that spans the entire $\Re^n$, we stop things short by picking $ms = n - \widetilde{O}(m)$ This set of extra columns significantly simplify the proof of singular value lower bounds. This is similar in spirit to the analysis of minimum singular values of random matrices, which is significantly easier for non-square matrices [RV10]. In the algorithm, the remaining columns are treated as a separate block that we reduce to via a Schur complement at the very end of the block elimination algorithm. Since the block is small, so is its overhead on the running time.

1.3 History and Related Work

Our algorithm has close connections with multiple lines of research on more efficient solvers for sparse linear systems. This topic has been extensively studied not only in computer science, but also in applied mathematics and engineering. For example, in the Editors of the Society of Industrial and Applied Mathematics News' 'top 10 algorithms of the 20th century', three of them (Krylov space methods, matrix decompositions, and QR factorizations) are directly related to linear systems solvers [Cip00].

At a high level, our algorithm is a hybrid linear systems solver. It combines iterative methods, namely block Krylov space methods, with direct methods that factorize the resulting Gram matrix of the Krylov space. Hybrid methods have their origins in the incomplete Cholesky method for speeding up elimination/factorization based direct solvers. A main goal of these methods is to reduce the $\Omega(n^2)$ space needed to represent matrix factorizations/inverses. This high space requirement is often even more problematic than time when handling large sparse matrices. Such reductions can occur in two ways: either by directly dropping entries from the (intermediate) matrices, or by providing more succinct representations of these matrices using additional structures.

The main structure of our algorithm is based on the latter line of work on solvers for structured matrices. Such systems arise from physical processes where the interactions between objects have invariances (e.g. either by time or space differences). Examples of such structure include circulant matrices [Gra06], Hankel/Toeplitz matrices [KKM79, BA80, XXG12, XXCB14], and distances from n-body simulations [CRW93]. Many such algorithms require exact preservation of the structure in intermediate steps. As a result, many of these works develop algorithms over finite fields [BA80, BL94, BJMS17].

More recently, there has been work on developing more numerically stable variants of these algorithms for structured matrices, or more generally, matrices that are numerically close to being structured [XCGL10, LLY11, XXG12, XXCB14]. However, these results only explicitly discussed the entry-wise Hankel/Toeplitz case (which corresponds to $s = 1$). Furthermore, because they rely on domain-decomposition techniques similar to fast multiple methods, they produce one bit of precision per each outer iteration loop. As the Krylov space matrix has condition number $\exp(\Omega(m))$, such methods would lead to another factor of m in the solve cost when directly invoked.

Instead, our techniques for handling and bounding numerical errors are more closely related to recent developments in provably efficient sparse Cholesky factorizations [KLP$^+$16, KS16, Kyn17, CKK$^+$18]. These methods generated efficient preconditioners using only the condition of intermediate steps of Gaussian eliminatnion, known as Schur complements, having small representations. They avoided the explicit generation of the dense representations of Schur complements by treatment them as operators, and implicitly applied randomized tools to directly sample/sketch the final succinct representations, which have much smaller algorithmic costs.

On the other hand, previous works on spare Choleskfy factorizations required the input matrix to be decomposable into a sum of simple elements, often through additional combinatorial structure of the matrices. In particular, this line of work on combinatorial preconditioning was initiated through a focus on graph Laplacians, which are built from 2-by-2 matrix blocks corresponding to edges of undirected graphs [Vai89, Gre96, ST14, KMP12]. Since then, there has been substantial generalizations to the structures amenable to such approaches, notably to finite element matrices [BHV08] and directed graphs/irreversible Markov chains [CKP$^+$17]. However, recent works have also shown that many classes of structures involving more than two variables are complete for general linear systems [Zha18]. Nonetheless, the prevalence of approximation errors in such algorithms led to the development of new ways of bounding numerical round-off errors in algorithms that are critical to our elimination routine for block-Hankel matrices.

Key to recent developments in combinatorial preconditioning is matrix concentration [RV07, Tro15]. Such bounds provide guarantees for (relative) eigenvalues of random sums of matrices. For generating preconditioners, such randomness arise from whether each element is kept, and a small condition number (which in turn implies a small number of outer iterations usign the preconditioners) corresponds to a small deviation between the original and sampled matrices. In contrast, we introduce randomness in order to obtain block Krylov spaces whose minimum eigenvalue is large. As a result, the matrix tool we need is anti-concentration, which somewhat surprisingly is far less studied. Previous works on it are mostly related by similar problems from numerical precision [SST06, TV10], and mostly address situations where the entries in the resulting matrix are independent. Our bound on the min singular value of the random Krylov space can yield a crude bound for a sum of rectangluar random matrices, but we believe much better matrix anti-concentration bounds are possible.

1.4 Organization

The rest of this paper is organized as follows: we present the “outer” algorithm in Section 2, and give a detailed outline of its analysis in Section 3. A breakdown of the main components of the analysis is in Section 3.2: briefly, Sections 4 and 5 bound the singular values of the block Krylov matrix, and Sections 6 and 7 give the linear systems solver with block Hankel matrices. Some research directions raised by this work, including possible improvements and extensions are discussed in Section 8.

2 Algorithm

We describe the algorithm, as well as the running times of its main components in this section. To simplify discussion, we assume the input matrix A is symmetric, and has $poly(n)$ condition number. If it is asymmetric (but invertible), we implicitly apply the algorithm to $A^T A$, using the identity $A^{-1} = (A^T A)^{-1} A^T$ derived from $(A^T A)^{-1} = A^{-1} A^{-T}$. Also, recall from the discussion after Theorem 1.1 that we use $\widetilde{O}(\cdot)$ to hide lower order terms in order to simplify runtimes.

Before giving details on our algorithm, we first discuss what constitutes a linear systems solver algorithm, specifically the equivalence between many such algorithms and linear operators.

For an algorithm $\textsc{Alg}$ that takes a matrix B as input, we say that $\textsc{Alg}$ is linear if there is a matrix $Z_{\textsc{Alg}}$ such that for any input B, we have

$$\textsc{Alg}\left(B\right) = Z_{\textsc{Alg}}.$$

In this section, in particular in the pseudocode in Algorithm 2, we use the name of the procedure, $\textsc{Solve}_A(b, \delta)$, interchangeably with the operator correpsonding to a linear algorithm that solves a system in A, on vector b, to error $\delta > 0$. In the more formal analysis, we will denote such corresponding linear operators using the symbol Z, with subscripts corresponding to the routine if appropriate.

This operator/matrix based analysis of algorithms was first introduced in the analysis of recursive Chebyshev iteration by Spielman and Teng [ST14], with credits to the technique also attributed to Rohklin. It the advantage of simplifying analyses of multiple iterations of such algorithms, as we can directly measure Frobenius norm differences between such operators and the exact ones that they approximate.

Under this correspondence, the goal of producing an algorithm that solves $Ax = b$ for any b as input becomes equivalent to producing a linear operator Z_A that approximates A^{-1}, and then running it on the input b. For convenience, we also let the solver take as input a matrix instead of a vector, in which case the output is the result of solves against each of the columns of the input matrix.

The high-level description of our algorithm is in Figure 2. To keep our algorithms as linear operators, we will ensure that the only approximate steps are from inverting matrices (where condition numbers naturally lead to matrix approximation errors), and in forming operators using fast convolution. We will specify explicitly in our algorithms when such round-off errors occur.

Some of the steps of the algorithm require care for, efficiency, as well as tracking the number of words needed to represent the numbers. We assume the bounds on bit-complexity in the analysis (Section 3) below, which is $\widetilde{O}(m)$ when $\kappa = poly(n)$, and use this in the brief description of costs in the outline of the steps below.

We start by perturbing the input matrix, resulting in a symmetric positive definite matrix where all eigenvalues are separated by α_A. Then we explicitly form a Krylov matrix from sparse Random Gaussians: For any vector u, we can compute $A^i u$ from $A^{i-1}u$ via a single matrix-vector multiplication in A. So computing each column of K requires $O(nnz(A))$ operations, each involving a length n vector with words of length $\widetilde{O}(m)$. So we get the matrix K, as well as AK, in time

$$\widetilde{O}\left(nnz\left(A\right) \cdot n \cdot m\right).$$

To obtain a solver for AK, we instead solve its Gram matrix $(AK)^T(AK)$. Each block of K^TK has the form $(G^S)^T A^i G^S$ for some $2 \leqslant i \leqslant 2m$, and can be computed by multiplying $(G^S)^T$ and A^iG^S. As A^iG^S is an n-by-s matrix, each non-zero in G^S leads to a cost of $O(s)$ operations involving words of length $\widetilde{O}(m)$. Then because we chose G^S to have $\widetilde{O}(m^3)$ non-zeros per column, the total number of non-zeros in G^S is about $\widetilde{O}(s \cdot m^3) = \widetilde{O}(nm^2)$. This leads to a total cost (across the m values of i) of:

$$\widetilde{O}\left(n^2m^3\right).$$

The key step is then Step 2: a block version of the Conjugate Gradient method. It will be implemented using a recursive data structure based on the notion of displacement rank [KKM79, BA80]. To get a sense of why a faster algorithm may be possible, note that there are only $O(m)$ distinct blocks in the matrix $(AK)^T(AK)$. So a natural hope is to invert these blocks by themselves: the cost of (stable) matrix inversion [DDH07], times the $\widetilde{O}(m)$ numerical word complexity, would then give a total of

$$\widetilde{O}\left(m^2 s^\omega\right) = \widetilde{O}\left(m^2\left(\frac{n}{m}\right)^\omega\right) = \widetilde{O}\left(n^\omega m^{\omega-2}\right).$$

Of course, it does not suffice to solve these m s-by-s blocks independently. Instead, the full algorithm, as well as the $\textsc{Solve}_M$ operator, is built from efficiently convolving such s-by-s blocks with matrices using Fast Fourier Transforms. Such ideas can be traced back to the development of super-fast solvers for (entry-wise) Hankel/Toeplitz matrices [BA80, LS92, XXCB14].

Choosing s and m so that $n = sm$ would then give the overal running time, **assuming that we can bound the minimum singular value of K by** $\exp(-\widetilde{O}(m))$. This is a

```
BlockKrylov( MatVec_A($x, \delta$): symmetric matrix given as implicit matrix vector
muliplication access, $\alpha_A$: eigenvalue range/separation bounds for $A$ that also
doubles as error threshold, $m$: Krylov step count )

  1. (FORM KRYLOV SPACE)
      (a) Set $s \leftarrow \lfloor n/m \rfloor - O(m)$, $h \leftarrow O(m^2 \log(1/\alpha_A))$. Let $G^S$ be an $n \times s$ random
          matrix with $G^S_{ij}$ set to $\mathcal{N}(0,1)$ with probability $\frac{h}{n}$, and 0 otherwise.
      (b) (Implicitly) compute the block Krylov space
              $K = \left[\ G^S \ \middle|\ AG^S \ \middle|\ A^2G^S \ \middle|\ \ldots \ \middle|\ A^{m-1}G^S\ \right].$
  2. (SPARSE INVERSE) Use fast solvers for block Hankel matrices to obtain a solver
     for the matrix:
              $M \leftarrow (AK)^T (AK)$,
     and in turn a solve to arbitrary error which we denote $\textsc{Solve}_M(\cdot, \epsilon)$.
  3. (PAD and SOLVE)
      (a) Let $r = n - ms$ denote the number of remaining columns. Generate a $n \times r$
          dense Gaussian matrix $G$, use it to complete the basis as: $Q = [K|G]$.
      (b) Compute the Schur complement of $(AQ)^T AQ$ onto its last $r = n - ms$ entries
          (the ones corresponding to the columns of $G$) via the operation
              $(AG)^T AG - (AG)^T \cdot AK \cdot \textsc{Solve}_M\left((AK)^T AG, \alpha_A^{10m}\right)$
          and invert this $r$-by-$r$ matrix.
      (c) Use the inverse of this Schur complement, as well as $\textsc{Solve}_M(\cdot, \epsilon)$ to obtain a
          solver for $Q^TQ$, $\textsc{Solve}_{Q^TQ}(\cdot, \epsilon)$.
  4. (SOLVE and UNRAVEL) Return the operator $Q \cdot \textsc{Solve}_{(AQ)^TAQ}((AQ)^T x, \alpha_A^{10m})$ as
     an approximate solver for $A$.
```

Figure 2: Pseudocode for block Krylov space algorithm: $\textsc{Solve}_{\cdot}(\cdot, \cdot)$ are operators corresponding to linear system solving algorithms whose formalization we discuss at the start of this section.

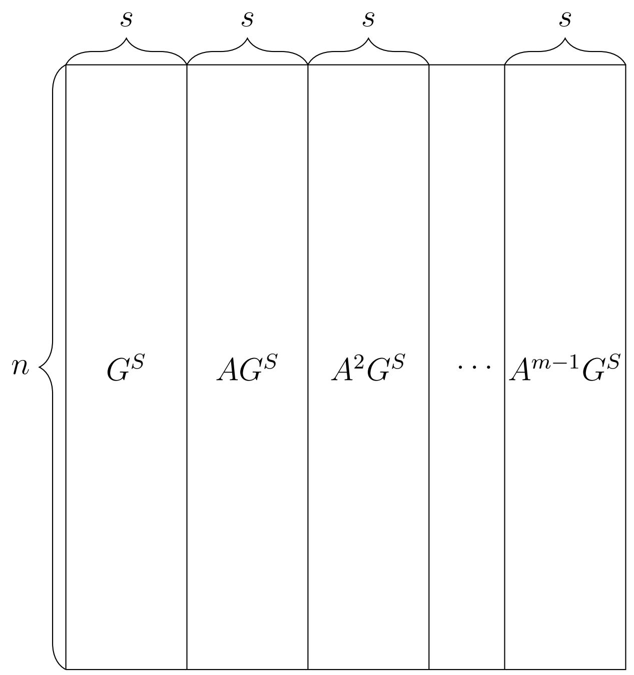

Figure 3: Randomized m-step Krylov Space Matrix with n-by-s sparse Gaussian G^S as starter.

major shortcoming of our analysis: we can only prove such a bound when $n - sm \geqslant \Omega(m)$. Its underlying cause is that rectangular semi-random matrices can be analyzed using ϵ-nets, and thus are significantly easier to analyze than square matrices.

This means we can only use m and s such that $n - ms = \Theta(m)$, and we need to pad K with $n - ms$ columns to form a full rank, invertible, matrix. To this end we add $\Theta(m)$ dense Gaussian columns to K to form Q, and solve the system AQ, and its associated Gram matrix $(AQ)^T(AQ)$ instead. These matrices are shown in Figure 4.

Because these additional columns are entry-wise i.i.d, its minimum singular value can be analyzed using existing tools [SST06, TV10], namely lower bounding the dot product of a random vector against any normal vector. Thus, we can lower bound the minimum singular value of Q, and in turn AQ, by $\exp(-\widetilde{O}(m))$ as well.

This bound in turn translates to the minimum eigenvalue of the Gram matrix of AQ, $(AQ)^T(AQ)$. Partitioning its entries by those from K and G gives four blocks: one (sm)-by-(sm) block corresponding to $(AK)^T(AK)$, one $\Theta(m)$-by-$\Theta(m)$ block corresponding to $(AG)^T(AG)$, and then the cross terms. To solve this matrix, we apply block-Gaussian elimination, or equivalently, form the Schur complement onto the $\Theta(m)$-by-$\Theta(m)$ corresponding to the columns in AG.

To compute this Schur complement, it suffices to solve the top-left block (corresponding to $(AK)^T(AK)$) against every column in the cross term. As there are at most $\Theta(m) < s$ columns, this solve cost comes out to less than $\widetilde{O}(s^\omega m)$ as well. We are then left with a

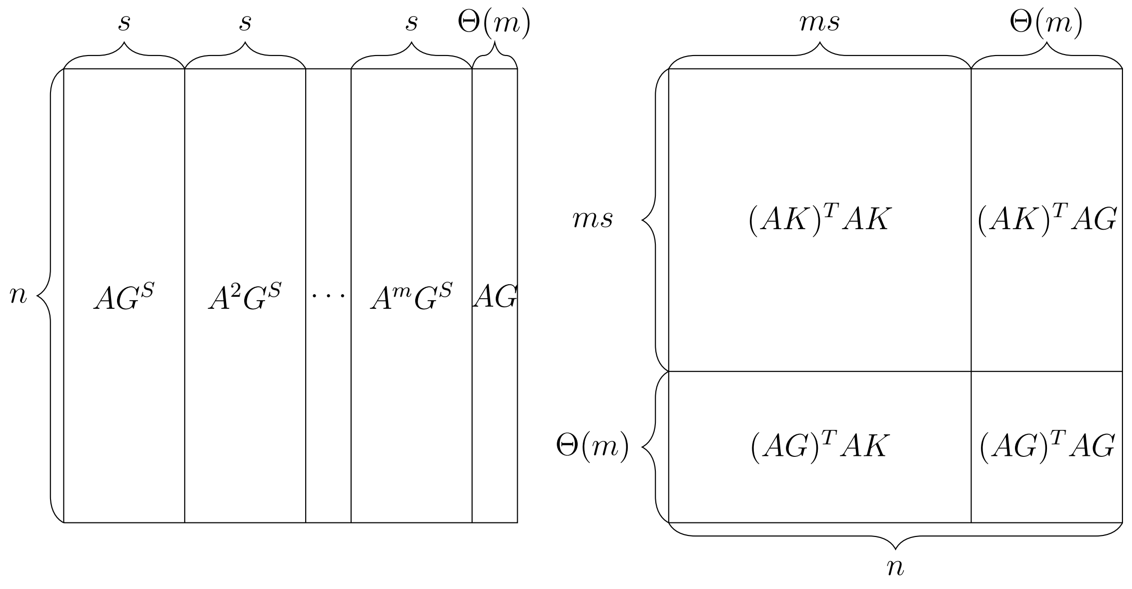

Figure 4: Full matrix AQ and its Associated Gram Matrix $(AQ)^T(AQ)$. Note that by our choice of parameters m is much smaller than $s \approx n/m$.

$\Theta(m)$-by-$\Theta(m)$ matrix, whose solve cost is a lower order term.

So the final solver operator costs

$$\widetilde{O}\left(nnz(A) \cdot nm + n^2m^3 + n^\omega m^{2-\omega}\right)$$

which leads to the final running time by choosing m to balance the terms. This bound falls short of the ideal case given in Equation 1 mainly due to the need for a denser B to the well-conditionedness of the Krylov space matrix. Instead of $O(n)$ non-zeros total, or about $O(m)$ per column, we need $poly(m)$ non-zero variables per column to ensure the an $\exp(-O(m))$ condition number of the block Krylov space matrix K. This in turn leads to a total cost of $O(n \cdot nnz \cdot poly(m))$ for computing the blocks of the Hankel matrix, and a worse trade off when summed against the $\frac{n^\omega}{m^{\omega-2}}$ term.

3 Outline of Analysis

In this section we outline our analysis of the algorithm through formal theorem statements. We start by formalizing our tracking of convergence, and the tracking of errors and roundoff errors.

3.1 Preliminaries

We will use capital letters for matrices, lower case letters for vectors and scalars. All subscripts are for indexing into entries of matrices and vectors, and superscripts are for indexing into entries of a sequence. Our notation is summarized in Table 1 at the end of the paper.

Norms and Singular Values. Our convergence bounds are all in terms of the Euclidean, or ℓ_2 norms. For a length n vector x, the norm of x is given by $\|x\| = \sqrt{\sum_{1 \leqslant i \leqslant n} x_i^2}$. Similarly, for a matrix M, the norm of its entries treated as a vector is known as the Frobenius norm, and we have

$$\|M\|_F = \sqrt{\sum_{ij} M_{ij}^2} = \sqrt{\textsc{Trace}\left(M^T M\right)}.$$

We will also use $\|\!|\cdot|\!\|$ to denote entry-wise norms over a matrix, specifically $\|M\|_\infty$ to denote the max magnitude of an entry in M. Note that $\|M\|_F = \|\!|M|\!\|_2$, so we have $\|\!|M|\!\|_\infty \leqslant \|M\|_F \leqslant n \|\!|M|\!\|_\infty$.

The minimum and maximum singular values of a matrix M are then defined as the min/max norms of its product against a unit vector:

$$\sigma_{\min}(M) = \min_x \frac{\|Mx\|_2}{\|x\|_2} \qquad \sigma_{\max}(M) = \max_x \frac{\|Mx\|_2}{\|x\|_2},$$

and the condition number of M is defined as $\kappa(M) = \sigma_{\max}(M)/\sigma_{\min}(M)$.

Bounds on the minimum singular value allows us to transfer perturbation errors to it to its inverse.

Lemma 3.1. *If M is a full rank square matrix with min and max singular values in the range $[\sigma_{\min}, \sigma_{\max}]$, and $\widetilde{M}$ is some approximation of it such that $\left\|\widetilde{M} - M\right\|_F \leqslant \epsilon$ for some $\epsilon < \sigma_{\min}/2$, then*

1. *All singular values in $\widetilde{M}$ are in the range $[\sigma_{\min} - \epsilon, \sigma_{\max} + \epsilon]$, and*
2. *The inverse of $\widetilde{M}$ is close to the inverse of M:*

$$\left\|\widetilde{M}^{-1} - M^{-1}\right\|_F \leqslant 10\sigma_{\min}^{-2}\epsilon.$$

Proof. The bound on singular values follows from the norm minimization/maximization definition of singular values. Specifically, we get that for a unit vector x,

$$\left|\left\|\widetilde{M}x\right\|_2 - \|Mx\|_2\right| \leqslant \left\|\left(\widetilde{M} - M\right)x\right\|_2 \leqslant \left\|\widetilde{M} - M\right\|_2 \|x\|_2 \leqslant \epsilon,$$

which means all singular values can change by at most ϵ.

Note that this implies that $\widetilde{M}$ is invertible. For the bounds on inverses, note that

$$\widetilde{M}^{-1} - M^{-1} = M^{-1}\left(M\widetilde{M}^{-1} - I\right) = M^{-1}\left(M - \widetilde{M}\right)\widetilde{M}^{-1}.$$

So applying bounds on norms, as well as $\left\|\widetilde{M}^{-1}\right\|_2 \leqslant (\sigma_{\min} - \epsilon)^{-1} \leqslant 2\sigma_{\min}^{-1}$ gives

$$\left\|\widetilde{M}^{-1} - M^{-1}\right\|_F \leqslant \left\|M^{-1}\right\|_2 \left\|M - \widetilde{M}\right\|_F \left\|\widetilde{M}^{-1}\right\|_2 \leqslant 2\sigma_{\min}^{-2}\epsilon.$$

□

Error Accumulation. Our notion of approximate operators also compose well with errors.

Lemma 3.2. *If $Z^{(1)}$ and $Z^{(2)}$ are linear operators with (algorithmic) approximations $\widetilde{Z}^{(1)}$ and $\widetilde{Z}^{(2)}$ such that for some $\epsilon < 0.1$, we have*

$$\left\|Z^{(1)} - \widetilde{Z}^{(1)}\right\|_F, \left\|Z^{(2)} - \widetilde{Z}^{(2)}\right\|_F \leqslant \epsilon$$

then their product satisfies

$$\left\|Z^{(1)}Z^{(2)} - \widetilde{Z}^{(1)}\widetilde{Z}^{(2)}\right\|_F \leqslant 10\epsilon \max\left\{1, \left\|Z^{(1)}\right\|_2, \left\|Z^{(2)}\right\|_2\right\}.$$

Proof. Expanding out the errors gives

$$Z^{(1)}Z^{(2)} - \widetilde{Z}^{(1)}\widetilde{Z}^{(2)} = \left(Z^{(1)} - \widetilde{Z}^{(1)}\right) Z^{(2)} + \left(Z^{(2)} - \widetilde{Z}^{(2)}\right) Z^{(1)} + \left(Z^{(1)} - \widetilde{Z}^{(1)}\right)\left(Z^{(2)} - \widetilde{Z}^{(2)}\right)$$

The terms involving the original matrix against the error gets bounded by the error times the norm of the original matrix. For the cross term, we have

$$\left\|\left(Z^{(1)} - \widetilde{Z}^{(1)}\right)\left(Z^{(2)} - \widetilde{Z}^{(2)}\right)\right\| \leqslant \left\|Z^{(1)} - \widetilde{Z}^{(1)}\right\|_F \cdot \left\|Z^{(2)} - \widetilde{Z}^{(2)}\right\|_2 \leqslant \epsilon^2,$$

which along with $\epsilon < 0.1$ gives the overall bound. □

Randomization and Normal Distributions Our algorithms rely on randomly perturbing the input matrices to make them non-degenerate, and much of our analysis revolving analyzing the effect of such perturbations on the eigenvalues. We make use of standard notions of probability, in particular, the union bound, which states that for any two events E_1 and E_2, $\Pr[E_1 \cup E_2] \leqslant \Pr[E_1] + \Pr[E_2]$.

Such a bound means that it suffices to show that the failure probability of any step of our algorithm is n^{-c} for some constant c. The total number of steps is $poly(n)$, so unioning over such probabilities still give a success probability of at least $1 - n^{-c+O(1)}$.

We will perturb our matrices using Gaussian random variables. These random variables $N(0, \sigma)$ have density function $g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-x^2/2\sigma^2}$. They are particularly useful for showing anti-concentration because the sum of Gaussians is another Gaussian, with variance equalling to the sum of squares, or ℓ_2^2-norm, of the variance terms. That is, for a vector x and a (dense) Gaussian vector with entry-wise i.i.d. $N(0, 1)$ normal random variables, aka. $g \sim N(0, 1)^n$, we have $x^T g \sim N(0, \|x\|_2)$.

The density function of Gaussians means that their magnitude exceed n with probability at most $O(\exp(-n^2))$. This probability is much smaller than the n^{-c} failure probabilities that we want, so to simplify presentation we will remove it at the start.

Claim 3.3. *We can analyze our algorithm conditioning on any normal random variable with variance σ, $N(0, \sigma)$, having magnitude at most $n\sigma$.*

Tracking Word Length. The numerical rounding model that we will use is fixed point precision. The advantage of such a fixed point representation is that it significantly simplifies the tracking of errors during additions/subtractions. The need to keep exact operators means we cannot omit intermediate digits. Instead, we track both the number of digits before and after the decimal point.

The number of trailing digits, or words after the decimal point, compound as follows:

1. Adding two numbers with L_1 and L_2 words after the decimal point each results in a number with $\max\{L_1, L_2\}$ words after the decimal point.

2. Multiplying two numbers with L_1 and L_2 words after the decimal point each results in a number with $L_1 + L_2$ words after the decimal point.

As $\max\{L_1, L_2\} \leqslant L_1 + L_2$ when L_1 and L_2 are non-negative, we will in general assume that when we multiply matrices with at most L_1 and L_2 words after the decimal point, the result has at most $L_1 + L_2$ words after the decimal point. In particular, if Z is an operator with L_Z words after the decimal point, and its input B has L_B words after the decimal point, the output has at most $L_Z + L_B$ words after the decimal point.

Note that both of these bounds are for exact computations. The only round off errors come from round-off errors by dropping some of the digits, as the matrices themselves are created.

On the other hand, we need to bound the maximum magnitude of our operators. The number of digits before the decimal point is given by bounds on the magnitude of the numbers themselves. Such bounds also propagate nicely along multiplications.

Lemma 3.4. *If the maximum magnitude of entries in two matrices Y and Z with dimension at most n are both at most α, then all entries in YZ have magnitude at most $n\alpha^2$ as well.*

Proof.

$$\left|(YZ)_{ij}\right| = \left|\sum_k Y_{ik} Z_{kj}\right| \leqslant \sum_k |Y_{ik}|\,|Z_{kj}| \leqslant n\alpha^2$$

□

Throughout our analyses, we will often rescale the matrices so that their max magnitudes are n^{-2}. This allows us to absorb any constant factor increases in magnitudes from multiplying these matrices by Lemma 3.4 because $(c \cdot n^{-2})^2 \leqslant n^{-2}$.

Finally, by doing FFT based fast multiplications for all numbers involved [CLRS09, HVDH19], we can multiple two numbers with an $O(\log n)$ factor in their lengths. This means that when handling two matrices with L_1 and L_2 words after the decimal point, and whose maximum magnitude is μ, the overhead caused by the word-lengths of the numbers involved is $\widetilde{O}(\mu + L_1 + L_2)$

Random Variables and Probability For our perturbations we use standard Gaussian $\mathcal{N}(0, 1)$ random variables.

Fact 3.5. *For $x \sim \mathcal{N}(0, 1)$, we have* $\Pr(|x| \geqslant t) \leqslant \frac{2}{t\sqrt{2\pi}} e^{-t^2/2}$.

Thus, with probability at least $1 - \exp(-n/2)$, a standard Gaussian variable is bounded by $\sqrt{n}$. We will use $O(n^2)$ such variables and condition on the event that all their norms are bounded by $\sqrt{n}$.

3.2 Main Technical Ingredients

The main technical components of the analysis can be summarized as follows:

1. A can be perturbed so that its eigenvalues are separated (Theorem 3.6, Section 4).
2. Such a separation implies a well-conditioned Krylov space, when it is initialized with sparse random Gaussian vectors. (Theorem 3.7, Section 5)
3. This Krylov matrix can be solved efficiently using a combination of low displacement rank solvers and fast convolutions (Theorem 3.8, Section 6).
4. The last few rows/columns can be solved efficiently via the Schur complement (Lemma 3.9, Section 7).

Anti-Concentration of semi-random matrices. A crucial part of our analysis is bounding the spectrum of semi-random matrices. Unfortunately, the highly developed literature on spectral properties of random matrices assumes independent entries or independent columns, which no longer hold in the semi-random case of K. However, getting tight estimates is not important for us (the running time is affected only by the logarithm of the gap/min value). So we adapt methods from random matrix theory to prove sufficient anti-concentration.

Specifically, after symmetrizing the potentially asymmetric input A by implicitly generating the operator $A^T A$, we need to bound (1) the minimum eigenvalue gap of the coefficient matrix after perturbation by a symmetric sparse matrix and (2) the minimum singular value of the block Krylov matrix constructed by multiplying with a sparse random matrix.

The first step of showing eigenvalue separation is needed because if A has a duplicate eigenvalue, the resulting Krylov space in it has rank at most $n-1$. We obtain such a separation by perturbing the matrix randomly: its analysis follows readily from recent results on separations of eigenvalues in random matrices by Luh and Vu [LV18]. In Section 4, we show the following separation bound.

Theorem 3.6. *For any $n \times n$ symmetric positive definite matrix $\overline{A}$ with:*

1. *entries at most $1/n$,*
2. *eigenvalues at least $1/\kappa$ for some $\kappa \geqslant n^3$,*

and any probability where

$$p \geqslant \frac{300 \log \kappa \log n}{n}$$

the symmetrically random perturbed matrix A defined as

$$A_{ij} = A_{ji} \stackrel{\text{def}}{=} \begin{cases} \bar{A}_{ij} + \frac{1}{n^2\kappa}\mathcal{N}(0,1) & w.p.\ p, \\ \bar{A}_{ij} & w.p.\ 1-p, \end{cases}$$

with probability at least $1-n^{-10}$ has all eigenvalues separated by at least $\kappa^{-5\log n}$.

Given this separation, we show that a random n-by-s B gives a m-step Krylov space matrix, as long as $n - ms = \Omega(m)$. Furthermore, we pick this B to be sparse in the columns, so we can quickly compute $B^T A^i B$.

Theorem 3.7. *Let A be an $n \times n$ symmetric positive definite matrix with entries at most $1/n$, and $\alpha_A < n^{-10}$ a parameter such that:*

1. *all eigenvalues of A are at least α_A, and*
2. *all pairs of eigenvalues of A are separated by at least α_A.*

Let s and m be parameters such that $n^{0.01} \leqslant m \leqslant n^{\frac{1}{4}}$ and $s \cdot m \leqslant n - 5m$. The n-by-s sparse Gaussian matrix G^S where each entry is set to $\mathcal{N}(0,1)$ with probability at least

$$\frac{10000 m^3 \log(1/\alpha_A)}{n}$$

leads to the Krylov space matrix

$$K = \left[\ G^S \ \middle|\ AG^S \ \middle|\ A^2G^S \ \middle|\ \ldots \ \middle|\ A^{m-1}G^S\ \right].$$

With probability at least $1-n^{-2}$ (over randomness in G^S), K has maximum singular value at most n^2, and minimum singular value at least α_A^{5m}.

We prove this in Section 5. These bounds allow us to bound the length of numbers, and in turn running time complexity of solving $(AK)^T AK$.

Solvers for block Hankel matrices. An important ingredient in our algorithm is a numerically efficient solver for block Hankel matrices. For this we use the notion of displacement rank by Kailath, Kung and Morf [KKM79]. Its key statement is that any Schur Complement of a Toeplitz Matrix has displacement rank 2, and can be uniquely represented as the factorization of a rank 2 matrix. This combined with the fact that the displacement rank of the inverse is the same as that of the matrix is used to compute the Schur complement in the first super-fast/asymptotically fast solvers for Toeplitz matrices by Bitmead and Anderson [BA80]. Here we extend this to block Toeplitz/Hankel matrices and prove numerical stability, using the natural preservation of singular values of Schur complements. Specifically, the analysis from Demmel, Dumitriu, Holtz and Kleinberg [DDHK07] can be readily adapted to this setting.

We give full details on the algorithm in Section 6.

Theorem 3.8. *If H is an $sm \times sm$ symmetric s-block-Hankel matrix and $0 < \alpha_H < (sm)^{-100}$ is a parameter such that every contiguous square block-aligned minor of H containing the top-right or bottom left corner have minimum eigenvalue at least α_H:*

$$\sigma_{\min}\left(H_{\{1:i,(m-i+1):m\}}\right), \sigma_{\min}\left(H_{\{(m-i+1):m,1:i\}}\right) \geqslant \alpha_H \qquad \forall 1 \leqslant i \leqslant m$$

and all entries in H have magnitude at most $(sm)^{-2}\alpha_H^{-1}$, then for any error ϵ, we can preprocess H in time $\widetilde{O}(ms^{\omega}\log(\alpha_H^{-1}\epsilon^{-1}))$ to form $\textsc{Solve}_H(\cdot,\epsilon)$ that corresponds to a linear operator Z_H such that:

1. *For any $(ms) \times k$ matrix B with max magnitude $\|\|B\|\|_\infty$ and L_B words after the decimal point, $\textsc{Solve}_H(B,\epsilon)$ returns $Z_H B$ in time*
$$\widetilde{O}\left(m \cdot \max\left\{s^{\omega-1}k, s^2k^{\omega-2}\right\} \cdot \left(\log\left(\frac{(1+\|\|B\|\|_\infty)\, ms}{\alpha_H \epsilon}\right) + L_B\right)\right).$$
2. *Z_H is a high-accuracy approximation to H^{-1}:*
$$\left\|Z_H - H^{-1}\right\|_F \leqslant \epsilon.$$
3. *the entries of Z_H have at most $O(\log^2 m \log(\alpha_H^{-1}\epsilon^{-1}))$ words after the decimal point.*

The overhead of $\log^2 m$ in the word lengths of Z_H is from the $O(\log n)$ layers of recursion in Fast Fourier Transform times the $O(\log n)$ levels of recursion in the divide-and conquer block Schur complement algorithm. Note that the relation between the i^{th} block of $H = K^T K$ and K itself is:

$$H_{\{1:i,(n-i+1):m\}} = K^T_{\{:,1:i\}} K_{\{:,(n-i+1):m\}} = K^T_{\{:,1:i\}} A^{m-i-1} K_{\{:,1:i\}}.$$

So the min/max singular values of these matrices off by a factor of at most α_A^m from the singular value bounds on H itself.

It remains to pad K with random columns to make it a square matrix, Q. We will bound the condition number of this padded matrix, and convert a solver for $M = (AK)^T(AK)$ to a solver for $(AQ)^T(AQ)$ in Section 7, Specifically, the following bounds are from combining Theorems 3.7 and 3.8, along with properties of random dense Gaussians.

Lemma 3.9. *Let A be an $n \times n$ symmetric positive definite matrix with entries at most $1/n$, and $0 < \alpha_A < n^{-10}$ a parameter such that:*

1. *all eigenvalues of A are at least α_A, and at most α_A^{-1},*
2. *all pairs of eigenvalues of A are separated by at least α_A,*
3. *all entries in A have magnitude at most α_A, and at most $O(\log(1/\alpha_A))$ words after the decimal point.*

For any parameter m such that $n^{0.01} \leqslant m \leqslant 0.01n^{0.2}$, the routine BlockKrylov *as shown in Figure 2 pre-processes A in time:*

1. *$O(n)$ matrix-vector multiplications of A against vectors with at most $O(m\log(1/\alpha_A))$ words both before and after the decimal point,*
2. *plus operations that cost a total of:*

$$\widetilde{O}\left(n^2 \cdot m^3 \cdot \log^2\left(1/\alpha_A\right) + n^{\omega}m^{2-\omega}\log\left(1/\alpha_A\right)\right).$$

and obtains a routine Solve$_A$ *that when given a vector b with L_b words after the decimal point, returns $Z_A b$ in time*

$$\widetilde{O}\left(n^2 m \cdot \left(\log\left(1/\alpha_A\right) + \left\|\left|b\right\|\right|_{\infty} + L_b\right)\right),$$

for some Z_A with at most $O(\log(1/\alpha_A))$ words after the decimal point such that

$$\left\|Z_A - A^{-1}\right\|_F \leqslant \alpha_A.$$

Note that we dropped ϵ as a parameter to simplify the statement of the guarantees. Such an omission is acceptable because if we want accuracy less than the eigenvalue bounds of A, we can simply run the algorithm with $\alpha_A \leftarrow \epsilon$ due to the pre-conditions holding upon α_A decreasing. With iterative refinement (e.g. [Saa03], it is also possible to lower the dependence on $\log(1/\epsilon)$ to linear instead of the cubic dependence on $\log(1/\alpha_H)$.

3.3 Proof of Main Theorem

It remains to use the random perturbation specified in Theorem 3.6 and taking the outer product to reduce a general system to the symmetric, eigenvalue well separated case covered in Lemma 3.9.

The overall algorithm then takes a symmetrized version of the original matrix $\overline{A}$, perturbs it, and then converts the result of the block Krylov space method back. Its pseudocode is in Figure 5

```
LinearEquationApproximation( $A$, $b$: integer matrix/vector pair, $\kappa$: condition number bound for $A$, $\epsilon$: error threshold. )

  1. Compute $\theta_A \leftarrow \|\|A\|\|_\infty$.
  2. Generate random symmetric matrix $R$ with
         $R_{ij} = R_{ji} = \frac{\epsilon}{n^{10}\kappa^2}\mathcal{N}(0,1)$
     with probability $\frac{O(\log(\kappa/\epsilon)\log n)}{n}$.
  3. Implicitly generate
         $\widetilde{A} = \frac{1}{n^4\theta_A^2}A^TA + R$
     and its associated matrix-multiplication operator $\textsc{MatVec}_{\widetilde{A}}(\cdot,\delta)$.
  4. Build solver for $\widetilde{A}$ via
         $\textsc{Solve}_{\widetilde{A}} \leftarrow \textsc{BlockKrylov}\left(\textsc{MatVec}_{\widetilde{A}}(\cdot,\delta), \left(n^8\kappa^2\epsilon^{-1}\right)^{-5\log n}, n^{\frac{\omega-2}{\omega+1}}nnz(A)^{\frac{\omega-2}{\omega+1}}\right)$.
  5. Return
         $\frac{1}{n^4\theta_A^2}\cdot\textsc{Solve}_{\widetilde{A}}\left(A^Tb\right)$.
```

Figure 5: Pseudocode for block Krylov space algorithm.

Proof. (Of Theorem 1.1) Let $\widehat{A}$ be the copy of A scaled down by $n^2\theta_A$:

$$\widehat{A} = \frac{1}{n^2\theta_A}A = \frac{1}{n^2\|A\|_\infty}A.$$

This rescaling gives us bounds on both the maximum and minimum entries of $\widehat{A}$. The rescaling ensures that the max magnitude of an entry in $\widehat{A}$ is at most $1/n^2$. Therefore its Frobenius norm, and in turn max singular value, is at most 1. On the other hand, the max singular of A is at least $\|\|A\|\|_\infty = \theta_A$: consider the unit vector that's 1 in the entry corresponding to the column containing the max magnitude entry of A, and 0 everywhere else. This plus the bound on condition number of κ gives that the minimum singular value of A is at least

$$\sigma_{\min}(A) \geqslant \frac{1}{\kappa}\sigma_{\max}(A) \geqslant \frac{\theta_A}{\kappa},$$

which coupled with the rescaling by $\frac{1}{n^2\theta_A}$ gives

$$\sigma_{\min}\left(\widehat{A}\right) \geqslant \frac{1}{n^2\theta_A}\cdot\frac{\theta_A}{\kappa} = \frac{1}{n^2\kappa}.$$

The matrix that we pass onto the block Krylov method, $\widetilde{A}$, is then the outer-product of $\widehat{A}$ plus a sparse random perturbation R with each entry is set (symmetrically when across the diagonal) to $\frac{\epsilon}{n^4\kappa}N(0,1)$ with probability $O(\log n \log(\kappa/\epsilon))/n$

$$\widetilde{A} \leftarrow \widehat{A}^T\widehat{A} + R.$$

This matrix $\widetilde{A}$ is symmetric. Furthermore, by Claim 3.3, we may assume that the max magnitude of an entry in R is at most $\epsilon n^{-9}\kappa^{-2}$, which gives

$$\|R\|_F \leqslant \frac{\epsilon}{n^8\kappa^2}.$$

So we also get that the max magnitude of an entry in $\widetilde{A}$ is still at most $2n^{-2}$. Taking this perturbation bound into Lemma 3.1 also gives that all eigenvalues of $\widetilde{A}$ are in the range

$$\left[\frac{1}{n^5\kappa^2}, 1\right].$$

By Theorem 3.6, the minimum eigenvalue separation in this perturbed matrix $\widetilde{A}$ is at least

$$\left(n^8\kappa^2\epsilon^{-1}\right)^{-5\log n}.$$

Also, by concentration bounds on the number of entries picked in R, its number of non-zeros is with high probability at most $O(n\log n\log(\kappa n/\epsilon)) = \widetilde{O}(n\log(\kappa/\epsilon))$.

As we only want an error of ϵ, we can round all entries in A to precision ϵ/κ without affecting the quality of the answer.

So we can invoke Lemma 3.9 with

$$\alpha_{\widetilde{A}} = \left(n^8\kappa^2\epsilon^{-1}\right)^{-5\log n},$$

which leads to a solve operator $Z_{\widetilde{A}}$ such that

$$\left\|Z_{\widetilde{A}} - \widetilde{A}^{-1}\right\|_F \leqslant \alpha_{\widetilde{A}} \leqslant \left(n^8\kappa^2\epsilon^{-1}\right)^{-5\log n} \leqslant \frac{\epsilon}{n^{40}\kappa^{10}}.$$

The error conversion lemma from Lemma 3.1 along with the condition that the min-singular value of $\widetilde{A}$ is at least $\frac{1}{n^5\kappa^2}$ implies that

$$\left\|Z_{\widetilde{A}}^{-1} - \widetilde{A}\right\|_F \leqslant \frac{\epsilon}{n^{30}\kappa^6}$$

or factoring into the bound on the size of R via triangle inequality:

$$\left\|Z_{\widetilde{A}}^{-1} - \widehat{A}^T\widehat{A}\right\|_F \leqslant \frac{2\epsilon}{n^8\kappa^4},$$

which when inverted again via Lemma 3.1 and the min singular value bound gives

$$\left\|Z_{\widetilde{A}} - \left(\widehat{A}^T\widehat{A}\right)^{-1}\right\|_F \leqslant \frac{\epsilon}{n^2}.$$

It remains to propagate this error across the rescaling in Step 5. Since $\widehat{A} = \frac{1}{n^2\theta_A}A$, we have

$$\left(A^TA\right)^{-1} = \frac{1}{n^4\theta_A^2}\left(\widehat{A}^T\widehat{A}\right)^{-1},$$

and in turn the error bound translates to

$$\left\|\frac{1}{n^4\theta_A^2}Z - \left(\widehat{A}^T\widehat{A}\right)^{-1}\right\|_F \leqslant \frac{\epsilon}{n^4\theta_A^2}.$$

The input on the other hand has

$$\Pi_A b = A\left(A^TA\right)^{-1}A^Tb,$$

so the error after multiplication by A is

$$A\left(\frac{1}{n^4\theta_A^2}Z - \left(\widehat{A}^T\widehat{A}\right)^{-1}\right)A^Tb,$$

which incorporating the above, as well as $\|A\|_2 \leqslant n\theta_A$ gives

$$\left\|A\left[\frac{1}{n^4\theta_A^2}ZA^Tb\right] - \pi_A b\right\|_2 \leqslant \frac{\epsilon}{n^3\theta_A}\left\|A^Tb\right\|_2.$$

On the other hand, because the max eigenvalue of A^TA is at most $\|A\|_F^2 \leqslant n^2\theta_A^2$, the minimum eigenvalue of $(A^TA)^{-1}$ is at least θ_A^{-2}. So we have

$$\|\Pi_A b\|_2 = \left\|A^Tb\right\|_{(A^TA)^{-1}} \geqslant \frac{1}{n^2\theta_A}\left\|A^Tb\right\|_2.$$

Combining the two bounds then gives that the error in the return value is at most $\epsilon\|\Pi_A b\|_2$.

For the total running time, the number of non-zeros in R implies that the total cost of mutliplying $\widetilde{A}$ against a vector with $\widetilde{O}(m\log(1/\alpha_A)) = \widetilde{O}(m\log n\log(\kappa/\epsilon)) = \widetilde{O}(m\log(\kappa/\epsilon))$ is

$$\widetilde{O}\left(\left(nnz\left(A\right)+n\right)m\log^2\left(\kappa/\epsilon\right)\right) \leqslant \widetilde{O}\left(nnz\left(A\right)m\log^2\left(\kappa/\epsilon\right)\right),$$

where the inequality of $nnz(A) \leqslant n$ follows pre-processing to remove empty rows and columns. So the total construction cost given in Lemma 3.9 simplifies to

$$O\left(n^2m^3\log^2\left(\kappa/\epsilon\right) + n^\omega m^{2-\omega}\log\left(\kappa/\epsilon\right) + n\cdot nnz\left(A\right)\cdot m\log\left(\kappa/\epsilon\right)\right)$$

The input vector, $\frac{1}{\theta_Y}y$ has max magnitude at most 1, and can thus be rounded to $O(\log(\kappa/\epsilon))$ words after the decimal point as well. This then goes into the solve cost with $\log(\||b\||_\infty) + L_b \leqslant O(\log(n\kappa/\epsilon))$, which gives a total of $\widetilde{O}(n^2m\log(\kappa/\epsilon))$, which is a lower order term compared to the construction cost. The cost of the additional multiplication in A is also a lower order term.

Optimizing m in this expression above based on only n and $nnz(A)$ gives that we should choose m so that

$$\max \left\{ n \cdot nnz\left(A\right) m, n^2 m^3 \right\} = n^{\omega} m^{2-\omega},$$

or

$$\max \left\{ n \cdot nnz\left(A\right) m^{\omega-1}, n^2 m^{\omega+1} \right\} = n^{\omega}.$$

The first term implies

$$m \leqslant \left(n^{\omega-1} \cdot nnz\left(A\right)^{-1}\right)^{\frac{1}{\omega-1}} = n \cdot nnz\left(A\right)^{\frac{-1}{\omega-1}}$$

while the second term implies

$$m \leqslant n^{\frac{\omega-2}{\omega+1}}.$$

Substituting the minimum of these two bounds back into $n^{\omega} m^{2-\omega}$ and noting that $2-\omega \leqslant 0$ gives that the total runtime dependence on n and $nnz(A)$ is at most

$$n^{\omega} \cdot \max \left\{ n^{\frac{(\omega-2)(2-\omega)}{\omega+1}}, n^{2-\omega} \cdot nnz\left(A\right)^{\frac{-(2-\omega)}{\omega-1}} \right\} = \max \left\{ n^{\frac{5\omega-4}{\omega+1}}, n^2 \cdot nnz\left(A\right)^{\frac{\omega-2}{\omega-1}} \right\}.$$

Incorporating the trailing terms, then gives the the bound stated in Theorem 1.1, with c set to 2 plus the number of log factors hidden in the $\widetilde{O}$. $\square$

4 Separating Eigenvalues via Perturbations

In this section we show that a small symmetric random perturbation to A creates a matrix whose eigenvalues are well separated. The main result that we will prove is Theorem 3.6, which we restate below.

Theorem 3.6. *For any $n \times n$ symmetric positive definite matrix $\overline{A}$ with:*

1. *entries at most $1/n$,*
2. *eigenvalues at least $1/\kappa$ for some $\kappa \geqslant n^3$,*

and any probability where

$$p \geqslant \frac{300 \log \kappa \log n}{n}$$

the symmetrically random perturbed matrix A defined as

$$A_{ij} = A_{ji} \stackrel{\text{def}}{=} \begin{cases} \bar{A}_{ij} + \frac{1}{n^2\kappa}\mathcal{N}\left(0,1\right) & w.p.\ p, \\ \bar{A}_{ij} & w.p.\ 1-p, \end{cases}$$

with probability at least $1 - n^{-10}$ has all eigenvalues separated by at least $\kappa^{-5\log n}$.

Our proof follows the outline and structure of the bounds on eigenvalue separations in random matrices by Nguyen, Tao and Vu [NTV17] for the dense case, and by Luh and Vu [LV18] and Lopatto and Luh [LL19] for the sparse case. The separation of eigenvalues needed for Krylov space methods was mentioned as a motivating application in [NTV17]. Our proof can be viewed as a more basic variant of the proof on gaps of random sparse matrices by Lopatto and Luh [LL19]. Because we only need to prove a polylog bound on the number of digits, we do not need to associate the $O(\log n)$ levels of ϵ-nets using chaining type arguments.

On the other hand, we do need to slightly modify the arguments in Lopatto and Luh [LL19] to handle the initial A matrix that's present in addition the random terms. These modifications we make are akin to the ones needed in the dense case by Nguyen, Tao and Vu [NTV17] to handle arbitrary expectations.

The starting point is the interlacing theorem, which relates the eigenvalue separation to the inner product of the last column with any eigenvector of the principle minor without it and its corresponding row, which we denote as $A-$.

Lemma 4.1. *Let A be a symmetric n-by-n matrix, and denote its $(n-1)$-by-$(n-1)$ leading minor by $A-$:*

$$A- = A_{1:(n-1),1:(n-1)}$$

Then the eigenavlues of A in sorted order, $\lambda(A,1) \leqslant \lambda(A,2) \leqslant \ldots \leqslant \lambda(A,n)$ and the eigenvalues of $A-$ in sorted order, $\lambda(A-,1), \leqslant \lambda(A-,2) \leqslant \ldots \leqslant \lambda(A-,n-1)$ interlace each other when aggregated:

$$\lambda\left(A,1\right) \leqslant \lambda\left(A-,1\right) \leqslant \lambda\left(A,2\right) \leqslant \ldots \lambda\left(A,n-1\right) \leqslant \lambda\left(A-,n-1\right) \leqslant \lambda\left(A,n\right).$$

In particular, this allows us to lower bound the gap between $\lambda_{i-1}(A)$ and $\lambda_i(A)$ by lower bounding the gap between the ith eigenvalues of A and $A-$. This bound can in turn be computed by left and right multiplying $A-$, the principle minor, against the corresponding eigenvectors of $A-$ and A.

Lemma 4.2. *Let $v(A,i)$ denote the eigenvector of $A-$ corresponding to $\lambda_i(A)$, and $v(A-,i)$ denote the eigenvector of $A-$ corresponding to $\lambda_i(A-)$. Then we have*

$$\left|\lambda_i\left(A\right) - \lambda_i\left(A-\right)\right| \geqslant \left|v\left(A,i\right)_n\right| \left|v\left(A-,i\right)^T A_{1:(n-1),n}\right|.$$

Proof. Taking the condition of $v(A,i)$ being an eigenvector for A:

$$Av\left(A,i\right) = \lambda\left(A,i\right) v\left(A,i\right)$$

and isolating it to the first $n-1$ rows gives:

$$\lambda\left(A,i\right) v\left(A,i\right)_{1:(n-1)} = \left[Av\left(A,i\right)\right]_{1:(n-1)} = \left(A-\right) v\left(A,i\right)_{1:(n-1)} + A_{1:(n-1),n} v\left(A,i\right)_n.$$

Left multiplying this equality by $v(A-,i)^T$, and invoking substituting in $v(A-,i)^T A- = \lambda(A-,i)v(A-,i)$ given by the fact that $v(A-,i)$ is an eigenvector for $A-$ gives:

$$\begin{aligned}\lambda(A,i) \cdot v(A-,i)^T v(A,i)_{1:(n-1)} \\ &= v(A-,i)^T (A-) v(A,i)_{1:(n-1)} + v(A-,i)^T A_{1:(n-1),n} v(A,i)_n \\ &= \lambda(A-,i) \cdot v(A-,i)^T v(A,i)_{1:(n-1)} + v(A,i)_n \cdot v(A-,i)^T A_{1:(n-1),n}.\end{aligned}$$

Moving the two terms involving the dot product $v(A-,i)^T v(A,i)_{1:(n-1)}$ onto the same side then gives an expression involving the difference of eigenvalues:

$$(\lambda(A,i) - \lambda(A-,i)) \cdot v(A-,i)^T v(A,i)_{1:(n-1)} = v(A,i)_n \cdot v(A-,i)^T A_{1:(n-1),n}.$$

or upon taking absolute values and dividing by the dot product

$$|\lambda(A,i) - \lambda(A-,i)| = \frac{\left|v(A,i)_n \cdot v(A-,i)^T A_{1:(n-1),n}\right|}{\left|v(A-,i)^T v(A,i)_{1:(n-1)}\right|}.$$

As both $v(A-,i)$ and $v(A,i)$ are unit vectors, we can also upper bound the denominator above by 1:

$$\left|v(A-,i)^T v(A,i)_{1:(n-1)}\right| \leqslant \|v(A-,i)\|_2 \left\|v(A,i)_{1:(s-1)}\right\|_2 \leqslant \|v(A-,i)\|_2 \|v(A,i)\|_2 \leqslant 1,$$

which then gives the lower bound on eigengap by the dot product of $v(A-,i)$ against the last column of A minus its last entry. □

Note that because $v(A,i)$ is unit lengthed, it has at least one entry with magnitude at least $n^{-1/2}$. By permutating the rows and columns, we can choose this entry to be entry n when invoking Lemma 4.1. So it suffices to lower bound the dot product of every column of A against the eigenvectors of the minor with that row/column removed. When the columns are dense Gaussians, this is simply a Gaussian whose variance is the norm of the eigenvector, which is 1.

For the sparse case, we need to prove that the eigenvector is dense in many entries. We will invoke the following lemma for every principle $(n-1)$-by-$(n-1)$ minor of A.

Lemma 4.3. *Let $\bar{A}$ and A be the initial and perturbed matrix as given in Theorem 3.6. That is, $\bar{A}$ is symmetric, has all entries with magnitude at most $1/n$, and minimum eigenvalue at least $1/\kappa$ for some $\kappa \geqslant n^3$; and A is formed by adding a symmetric sparse Gaussian matrix where each entry is set to $\frac{1}{n^2\kappa} \cdot \mathcal{N}(0,1)$ with probability $300 \log n \log \kappa / n$, and 0 otherwise. Then with probability at least $1 - n^{-20}$, all eigenvectors of A have at least $\frac{n}{40 \log n}$ entries with magnitude at least $\kappa^{-3 \log n}$.*

The proof is by inductively constructing $O(\log n)$ levels of ϵ nets. A similar use of this technique for bounding the density of null space vectors in matrices with completely independent entries is in Lemma 5.6.

Proof. First, by Claim 3.3, we may assume that all entries in the perturbation generated are bounded by $1/(\kappa n) \leqslant \frac{1}{n}$. This in turn gives $\|A - \bar{A}\|_F \leqslant 1/(n\kappa)$, and so by Lemma 3.1, all eigenvalues of A are between $\frac{1}{2\kappa}$ and 2.

We will prove by induction on i that for all $i \in [0, \log(\frac{n}{1000 \log n})]$, with probability at least $1 - i \cdot n^{-21}$, all unit length vectors x such that $Ax = \lambda x$ for some λ with absolute value in the range $[\frac{1}{2\kappa}, 2]$ has at least 2^i entries with magnitude at least $\kappa^{-100(i+1)}$.

The base case of $i = 0$ follows that the norm of x being 1: at least one of n entries in x must have magnitude at least $1/n$.

For the inductive case, we construct an ϵ-net consisting of all vectors with entries that are integer multiples of ϵ_i, which we set to

$$\epsilon_i \leftarrow \kappa^{-3(i+1)}.$$

For any vector x, rounding every entry in x to the nearest multiple of ϵ_i produces a vector $\widehat{x}$ such that

$$\|x - \widehat{x}\|_2 \leqslant \sqrt{n}\epsilon_i$$

which when multiplied by the norm of A gives

$$\|Ax - A\widehat{x}\|_2 \leqslant \|A\|_2 \cdot \|x - \widehat{x}\|_2 \leqslant 2\sqrt{n}\epsilon_i.$$

Combining these then gives

$$\|A\widehat{x} - \lambda\widehat{x}\|_2 \leqslant \|A - \lambda x\| + 4\sqrt{n}\epsilon_i.$$

This means it suffices to show that with good probability, for all $\widehat{x}$ with:

1. at most 2^i non-zero entries,
2. all non-zeros are integer multiple of ϵ_i,
3. overall norm at most 2,

there does not exist some λ with $|\lambda| \in [\frac{1}{2\kappa}, 2]$ such that

$$\|A\widehat{x} - \lambda\widehat{x}\|_2 \leqslant 10\sqrt{n}\epsilon_i.$$

By the inductive hypothesis, it suffices to consider x with at least 2^{i-1} entries with magnitude at least ϵ_{i-1}. Such vectors in turn round to $\widehat{x}$ with at least 2^{i-1} entries whose magnitudes are at least

$$\epsilon_{i-1} - \epsilon_i \geqslant \frac{1}{2}\epsilon_{i-1}.$$

Let this subset of entries be Large.

We will bound the probability of $\widehat{x}$ being close to an eigenvector by applying the approximation condition to its zeros. The choice of i so that $2^i \leqslant \frac{n}{1000 \log n} \leqslant \frac{n}{2}$ means $\widehat{x}$

has at least $n/2$ entries that are 0. For each such entry i, the fact that $\lambda \widehat{x}_i = 0$ for any choice of λ means it suffices to upper bound the probability over A of

$$|A_{i,:}\widehat{x}| \leqslant 10\sqrt{n}\epsilon_i$$

As these entries with zeros are disjoint from the ones in LARGE, the corresponding entries in A are independent from each other. So we can consider the rows corresponding to each such i independently. The probability that no entry in $A_{i,\text{LARGE}}$ is chosen to be a non-zero is

$$(1-p)^{2^{i-1}} = \left(1 - \frac{300 \log n \log \kappa}{n}\right)^{2^{i-1}} \geqslant \exp\left(\frac{-300 \log n \log \kappa 2^{i-1}}{n}\right).$$

On the other hand, if one such entry is picked, the resulting Gaussian corresponding to $A_{i,:}\widehat{x}$ has variance at least

$$\frac{\epsilon_{i-1}}{2} \cdot \frac{1}{n^2 \kappa},$$

so its probability of being in an interval of size at most $10\sqrt{n}\epsilon_i$ is at most

$$10\sqrt{n}\epsilon_i \frac{2n^2\kappa}{\epsilon_{i-1}} = 20n^{2.5}\kappa \frac{\epsilon_i}{\epsilon_{i-1}} \leqslant \kappa^{-1},$$

where the second inequality uses $\epsilon_{i+1} = \kappa^{-3}\epsilon_i$ and $\kappa \geqslant n^3$. Combining these two then gives

$$\mathrm{Pr}_A\left[|A_{i,:}\widehat{x}| \leqslant n^2\epsilon_i\right] \leqslant \max\left\{\exp\left(\frac{-300 \log \kappa 2^{i-1}}{n}\right), \kappa^{-1}\right\},$$

which compounded over the at least $n/2$ choices of i gives that the overall probability of A being picked so that $\widehat{x}$ could be close to an eigenvector is at most

$$\max\left\{\exp\left(-300 \log \kappa 2^{i-2}\right), \kappa^{-n/2}\right\}.$$

So it remains to take a union bound of this probability over the size of the ϵ-net. Recall that we considered all $\widehat{x}$ with at most 2^i nonzeros, and each such non-zero is an integer multiple of ϵ_i in the range $[-2, 2]$. This means the total size of the ϵ-net can be upper bounded by:

$$\binom{n}{2^i} \cdot \left(\frac{4}{\epsilon_i}\right)^{2^i} \leqslant n^{2^i} \cdot \kappa^{3(i+1)\cdot 2^i} \leqslant \kappa^{7i \cdot 2^i},$$

where the last inequality uses the assumption of $i \geqslant 1$.

We then invoke union bound over this ϵ net. That is, we upper bound the overall failure probability by multiplying the two failures probabilities for individual vectors obtained earlier against this size bound above. For the first term, we get

$$\kappa^{7 \cdot i \cdot 2^i} \cdot \exp\left(-300 \log \kappa \log n 2^{i-2}\right) = \kappa^{7i2^i - 300 \log n 2^{i-2}},$$

which is less than n^{-22} because $i < \log n$.

For the second term, we get

$$\kappa^{7i \cdot 2^i} \cdot \kappa^{-n/2} = \kappa^{7 \cdot i \cdot 2^i - n/2},$$

which is at most $\kappa^{-22} \leqslant n^{-22}$ when $2^i \leqslant \frac{n}{20 \log n}$. Thus the overall failure probability is at most $2 \cdot n^{-22} \leqslant n^{-21}$, and the inductive hypothesis holds for i as well by union bound.

The choice of i being powers of 2s means that the last value picked is at least half the upper bound, aka. at least $\frac{n}{40 \log n}$. So we get that with probability at least $1 - n^{-20}$, at least this many entries have entries have magnitude at least $\kappa^{-3 \log n}$. □

Applying Lemma 4.3 to every principle minor that remove one row and column of A, and incorporating the dot product based lower bound on eigenvalue gap from Lemma 4.2 then gives the overall bound.

Proof. (of Theorem 3.6) First, consider all the s principle minors of A formed by removing one row and column (which we denote by k),

$$A_{[n] \setminus k, [n] \setminus k}.$$

In order to invoke Lemma 4.2, we need to first lower bound the value of column k when multiplied by all the eigenvectors of this principle minor,

$$\left| A_{[n] \setminus k, k}^T v \left(A_{[n] \setminus k, [n] \setminus k}, i \right) \right|.$$

Note that the entries in this matrix are also perturbed in the same manner as the overall matrix, Therefore, Lemma 4.3 gives that with probability at least $1 - n^{-20}$, for each i, all entries of $v(A_{[n] \setminus k, [n] \setminus k}, i)$ $\frac{n}{40 \log n}$ entries with magnitude at least $\kappa^{-3 \log n}$.

On the other hand, the entries of $A_{[n] \setminus k}$ are each perturbed by $\frac{1}{n^2 \kappa} \mathcal{N}(0, 1)$ with probability at least $\frac{30 \log \kappa \log n}{n}$. This means the probability of one of these perturbations occurring on a large magnitude entry of $v(A_{[n] \setminus k, [n] \setminus k}, i)$ is at least

$$\begin{aligned} 1 - \left(1 - \frac{300 \log \kappa \log n}{n} \right)^{\frac{n}{40 \log n}} &\geqslant 1 - \exp \left(- \frac{300 \log \kappa \log n}{n} \cdot \frac{n}{40 \log n} \right) \\ &= 1 - \exp \left(- \frac{300 \log \kappa}{40} \right) \geqslant 1 - n^{-20}, \end{aligned}$$

where the last inequality follows from $\kappa \geqslant n^3$. Thus, we have that with probability at least $1 - n^{-20}$, the variance in the dot product between $A_{[n] \setminus k, k}$ and $v(A_{[n] \setminus k, [n] \setminus k}, i)$ is at least

$$\kappa^{-3 \log n} \cdot \frac{1}{n^2 \kappa},$$

which in turn implies with probability at least $1 - n^{-20}$, this dot product is at least $\kappa^{-4 \log n}$ in magnitude.

Taking a union bound over all n choices of k, and all $n-1$ eigenvectors of $A_{[n]\setminus k,[n]\setminus k}$ gives that with probability at least $1-n^{-10}$, we have

$$\left|A^T_{[n]\setminus k,k} v(A_{[n]\setminus k,[n]\setminus k}, i)\right| \geqslant \kappa^{-4\log n} \qquad \forall k, i.$$

Then consider any $v(A,i)$. Since it has norm 1, there is some entry k with $|v(A,i)_k| \geqslant n^{-1/2}$. Plugging this into Lemma 4.2 gives:

$$\left|\lambda\left(A_{[s]\setminus k,[s]\setminus k}, i\right) - \lambda\left(A, i\right)\right| \geqslant n^{-1/2} \cdot \kappa^{-4\log n} \geqslant \kappa^{-5\log n}.$$

Combining this with interlacing from Lemma 4.1, namely $\lambda(A,i) \leqslant \lambda(A_{[n]\setminus k,[n]\setminus k}, i) \leqslant \lambda(A, i+1)$ then lower bounds the gap between $\lambda_i(A,i)$ and $\lambda(A,i+1)$ by the same amount. □

5 Condition Number of a Random Krylov Space

In this section, we use ϵ-nets to bound the condition number of a Krylov space matrix generated from a random sparse matrix.

Theorem 3.7. *Let A be an $n \times n$ symmetric positive definite matrix with entries at most $1/n$, and $\alpha_A < n^{-10}$ a parameter such that:*

1. *all eigenvalues of A are at least α_A, and*
2. *all pairs of eigenvalues of A are separated by at least α_A.*

Let s and m be parameters such that $n^{0.01} \leqslant m \leqslant n^{\frac{1}{4}}$ and $s \cdot m \leqslant n - 5m$. The n-by-s sparse Gaussian matrix G^S where each entry is set to $\mathcal{N}(0,1)$ with probability at least

$$\frac{10000 m^3 \log\left(1/\alpha_A\right)}{n}$$

leads to the Krylov space matrix

$$K = \left[\ G^S \ \middle|\ AG^S \ \middle|\ A^2 G^S \ \middle|\ \ldots \ \middle|\ A^{m-1} G^S\ \right].$$

With probability at least $1-n^{-2}$ (over randomness in G^S), K has maximum singular value at most n^2, and minimum singular value at least α_A^{5m}.

Our overall proof structure builds upon the eigenvalue lower bounds for random matrices with independent entries [SST06, TV10]. This approach is also taken in Lemma 6.11 (with suboptimal parameter trade-offs) to analyze the min singular value of a matrix perturbed by dense Gaussian matrices.

The main difficulty we need to address when adapting this approach is the dependence between columns of the Krylov space matrix. For a particular i, the columns of $A^i G^S$ are still independent, but for a single column of G^S, g^S and two different powers i_1

and i_2, $A^{i_1} g^S$ and $A^{i_2} g^S$ are dependent. Such dependencies requires us to analyze the m columns generated from each column of G^S together. Specifically, the m columns $g^S, Ag^S, \ldots, A^{m-1}g^S$ produced by each such vector g^S.

We then consider the space orthogonal to rest of the Krylov space, which we denote using W. Our matrix generalization for showing that the columns corresponding to g^S having large projection into W is to lower bound the right singular values of the matrix

$$W^T \left[g^S, Ag^S, \ldots, A^{m-1}g^S\right],$$

or equivalently, showing that its products against all length m unit vectors have large norms.

The condition of $m \times s \leqslant n - 5m$ represents the key that allows the use of ϵ nets. It implies that W has at least $5m$ rows, and thus the randomness in g^S has more dimensions to show up than the dimensions of length m unit vectors. In Section 5.1, we transform the product for some $\widehat{x}$ in the ϵ net,

$$W^T \left[g^S, Ag^S, \ldots, A^{m-1}g^S\right] \widehat{x}$$

into the product of a matrix with high numerical rank (about $\Omega(m)$) against the non-zero entries of g^S. This is done through the connection between Krylov spaces and Vandermonde matrices, which we formalize in the rest of this section and Appendix A.

However, for even a single vector to work, we need to ensure that the columns of W are not orthogonal to $A^i g^S$. Consider the following example: A is diagonal, and all the non-zeros in the columns of W are in about $O(m)$ rows. Then $A^i g^S$ is non-zero only if the non-zeros of g^S overlap with the non-zero rows of W. As we choose entry of g^S is set to non-zero with probability about m^3/n, this overlap probability works out to about $O(m^4/n) < 0.1$. In other words, $W^T[g^S, Ag^S, \ldots A^{m-1}g^S]$ would be 0 with constant probability.

Therefore, we need to rule out W that are 'sparse'. For A with general eigenspace structures, this condition becomes any vector in the column space of W having large dot products with many (about $n/poly(m)$) eigenvectors of A. We will formalize the meaning of such density through the spectral decomposition of A in the rest of this section. Then in Section 5.2 we use a multi-layer ϵ-net argument similar to the eigenvector density lower bound from Lemma 4.3 to conclude that any vector in the column span of W has dense projections in the eigenspaces.

We start by taking the problem into the eigenspaces of A, and consider the effect of each column of G^S as a column block. Since A is symmetric, let its spectral decomposition be

$$A = U^T \mathrm{DIAG}\left(\sigma\right) U$$

where σ is the list of eigenvalues. Then for a single column g^S, the resulting vectors are

$$\left[\ U^T U g^S \ \middle|\ U^T \mathrm{DIAG}\left(\sigma\right)^1 U g^S \ \middle|\ U^T \mathrm{DIAG}\left(\sigma\right)^2 U g^S \ \middle|\ \ldots \ \middle|\ U^T \mathrm{DIAG}\left(\sigma\right)^{m-1} U g^S \ \right].$$

Since U^T is a unitary matrix, we can remove it from consideration. Furthermore, note that for any vectors x and y,

$$\mathrm{DIAG}\,(x)\, y = \mathrm{DIAG}\,(y)\, x$$

so we can switch the roles of σ and Ug^S, by defining the Vandermonde matrix $V \in \Re^{n \times m}$ with entries given by:

$$V_{ij} = \sigma_i^{j-1}.$$

With this matrix, the term corresponding to g^S can be written as

$$\mathrm{DIAG}\left(Ug^S\right) V.$$

More globally, we are now considering a matrix formed by taking s copies (one per column of G^S) of V with rows rescaled by DIAG $\left(Ug^S\right)$, and putting the columns beside each other.

As mentioned before, V is a Vandermonde matrix. Its key property that we will use is below. We will prove this for completeness in Appendix A.

Lemma 5.1. *Let σ be a length n vector with all entries in the range $\left[\alpha, \alpha^{-1}\right]$ for some $\alpha < 1/n$, and any two entries of σ at least α apart. The $n \times m$ Vandermonde matrix with entries given as*

$$V_{ij} = \sigma_i^{j-1}.$$

has the property that for any unit vector x, at least $n - m + 1$ entries in Vx have magnitude at least α^{3m}.

We will treat this property of V as a black box. Specifically, the main technical result that we will show using ϵ-nets is:

Lemma 5.2. *Let V be an $n \times m$ matrix with:*

1. *$m < n^{1/4}$,*
2. *max entry magnitude at most 1,*
3. *for any unit x, Vx has at least $n - m + 1$ entries with magnitude at least α_V for some $\alpha_V \leqslant n^{-5 \log n}$.*

and let U be an $n \times n$ orthonormal matrix. Let G^S be an $n \times s$ sparse Gaussian matrix where each entry is set i.i.d. to $\mathcal{N}(0, 1)$ with probability h/n, where $s \geqslant \frac{n}{2m}$, $h \geqslant 10000 m^2 \log(1/\alpha_V)$, and W be an $n \times r$ orthonormal matrix orthogonal to G^S for some $r \geqslant 5m$, i.e., $W^T G^S = 0$.

Then for another column vector g^S generated from this sparse Gaussian distribution with density h, we have with probability at least $1 - n^{-3}$:

$$\sigma_{\min}\left(W^T \mathrm{DIAG}\left(Ug^S\right) V\right) \geqslant \alpha_V^4.$$

We will prove this bound in the next subsection (Section 5.1). Before we do so, we first formally verify that this local bound suffices for globally lower bounding the min singular value of the block Krylov space matrix.

Proof. (of Theorem 3.7)

Claim 3.3 allows us to assume that the max magnitude of an entry in G^S is at most n. Applying Lemma 3.4 inductively on $A^i G^S$ then gives that the magnitudes of all entries in each such matrix is at most n as well. Thus the max magnitude of an entry in K is at most n, and the max singular value of K is at most n^2.

We now turn attention to the min singular value. Since U is unitary, it suffices to lower bound the min singular vlaue of

$$\widehat{K} = UK = \textsc{HCat}_{1 \leqslant j \leqslant s}\left(\text{DIAG}\left(UG^S_{:,j}\right)V\right).$$

where $\textsc{HCat}(\cdot)$ denotes horizontal concatenation of matrices (with the same numbers of rows), and V is m-step Vandermonde matrix generated from the eigen values of A. Since the Frobenius norm of A is at most 1, all eigenvalues, and all their powers, have magntiude at most 1 as well. Combining this with Lemma 5.1 then gives that V has max entry-wise magnitude at most 1, and for any unit vector x, Vx has at least $n - m + 1$ entries with magnitude at least α_A^{3m}.

We denote the block of m columns corresponding to $G^S_{:,j}$ as $[j]$. Taking union bound over all column blocks, we get that Lemma 5.2 holds for each of the blocks. Under this assumption, we proceed to lower bound bound $\left\|\widehat{K}x\right\|_2$ for all unit vectors x. For each such vector, because there are at most s blocks, there is some block $x[j]$ with $\|x[j]\|_2 \geqslant s^{-1/2}$.

The energy minimization extension definition of Schur complements (e.g. Appendix A.5.5. of [BBV04]) gives

$$x^T\left(\widehat{K}^T\widehat{K}\right)x \geqslant x\left[j\right]^T \textsc{Sc}\left(\widehat{K}^T\widehat{K},[j]\right)x\left[j\right]$$

This Schur complement of $\widehat{K}^T\widehat{K}$ onto the block $[j]$ can in turn be written as

$$\widehat{K}^T_{:,[j]}\widehat{K}_{:,[j]} - \widehat{K}^T_{:,[j]}\widehat{K}_{:,\overline{[j]}}\left(\widehat{K}^T_{:,\overline{[j]}}\widehat{K}_{:,\overline{[j]}}\right)^{\dagger}\widehat{K}^T_{:,\overline{[j]}}K_{:,[j]} = \widehat{K}^T_{:,[j]}\left(I - \widehat{K}_{:,\overline{[j]}}\left(\widehat{K}^T_{:,\overline{[j]}}\widehat{K}_{:,\overline{[j]}}\right)^{\dagger}\widehat{K}^T_{:,\overline{[j]}}\right)\widehat{K}_{:,[j]}$$

where $\dagger$ denotes the pseudo-inverse.

Note that the middle term is precisely a projection matrix onto the space orthogonal to the columns of $K_{:,\overline{[j]}}$. So in particular, picking $W(\overline{[j]})$ to be an orthogonal basis of a subset of this space can only decrease the operator:

$$\textsc{Sc}\left(\widehat{K}^T\widehat{K},[j]\right) \succeq \widehat{K}^T_{:,[j]}W\left(\overline{[j]}\right)W\left(\overline{[j]}\right)^T\widehat{K}_{:,[j]}$$

Applying this matrix inequality to the vector $x[j]$ then gives:

$$\left\|\widehat{K}x\right\|_2 \geqslant \sqrt{x\left[j\right]^T \widehat{K}_{:,[j]}^T W\left(\overline{[j]}\right) W\left(\overline{[j]}\right)^T \widehat{K}_{:,[j]} x\left[j\right]} = \left\|W\left(\overline{[j]}\right)^T \widehat{K}_{:,[j]} x\left[j\right]\right\|_2$$
$$\geqslant \sigma_{\min}\left(W\left(\overline{[j]}\right)^T \widehat{K}_{:,[j]}\right) \cdot \left\|x\left[j\right]\right\|_2 \geqslant n^{-1/2} \cdot \sigma_{\min}\left(W\left(\overline{[j]}\right)^T \widehat{K}_{:,[j]}\right).$$

Then the result follows from the lower bound on the min singular value from Lemma 5.2.

□

5.1 Least Singular Value of Krylov Space of One Column

We now prove the lower bound on min singular value of a single block as stated in Lemma 5.2.

Lemma 5.2. *Let V be an $n \times m$ matrix with:*

1. *$m < n^{1/4}$,*
2. *max entry magnitude at most 1,*
3. *for any unit x, Vx has at least $n - m + 1$ entries with magnitude at least α_V for some $\alpha_V \leqslant n^{-5 \log n}$.*

and let U be an $n \times n$ orthonormal matrix. Let G^S be an $n \times s$ sparse Gaussian matrix where each entry is set i.i.d. to $\mathcal{N}(0,1)$ with probability h/n, where $s \geqslant \frac{n}{2m}$, $h \geqslant 10000m^2 \log(1/\alpha_V)$, and W be an $n \times r$ orthonormal matrix orthogonal to G^S for some $r \geqslant 5m$, i.e., $W^T G^S = 0$.

Then for another column vector g^S generated from this sparse Gaussian distribution with density h, we have with probability at least $1 - n^{-3}$:

$$\sigma_{\min}\left(W^T \text{DIAG}\left(Ug^S\right) V\right) \geqslant \alpha_V^4.$$

The proof will be via an ϵ-net argument. We want to show that for all $x \in \Re^m$, $\left\|W^T \text{DIAG}\left(Ug^S\right) Vx\right\|$ is large. To do so, we will enumerate over all vectors, with entries rounded to nearest multiple of ϵ. We will denote these rounded vectors with the hat superscript, i.e., $\widehat{x}$ for x.

By claim 3.3, we may assume that the entries in Ug^S have magnitude at most $2\sqrt{n}$. Combining this with the fact that W is orthonormal, and the given condition on max magnitude of V gives that for vectors x and $\widehat{x}$ such that $\|x - \widehat{x}\|_\infty \leqslant \epsilon$, we have

$$\left\|W^T \text{DIAG}\left(Ug^S\right) V\widehat{x} - W^T \text{DIAG}\left(Ug^S\right) Vx\right\| \leqslant n^3 \epsilon.$$

So we can consider a finite number of such $\widehat{x}$ vectors, as long as ϵ is set to be smaller than the lower bound on products we want to prove.

For such a vector $\widehat{x}$, we let

$$\widehat{y} = V\widehat{x}.$$

As $\frac{1}{2} \leqslant \|\widehat{x}\|_2 \leqslant 2$, the max magnitude of an entry in $\widehat{y}$ is at most $2n$ by the bound on magnitude of entries of V; and at least $n-m$ entries in $\widehat{y}$ have magnitude at least $\alpha_V/2$ by the given condition on V.

We want to show for a fixed triple of W^T, U, and $\widehat{y}$, it's highly unlikely for g^S to result in a small value of $\left\|W^T \mathrm{DIAG}\left(Ug^S\right)\widehat{y}\right\|$. Consider the effect of a single entry of g^S, g_i^S on the result. The vector produced by g_i^S in $U^T g^S$ is

$$\left(U^T\right)_{:,i} = \left(U_{i,:}\right)^T.$$

This vector is then multiplied entry-wise by $\widehat{y}$:

$$\mathrm{DIAG}\left(U_{i,:}^T\right)\widehat{y} = \mathrm{DIAG}\left(\widehat{y}\right)\left(U_{i,:}\right)^T,$$

and then multiplied against W^T. So the contribution of g_i^S to the overall sum is a vector in $\Re^k$ given by

$$g_i^S \cdot W^T \mathrm{DIAG}\left(\widehat{y}\right) U_{i,:}^T.$$

Thus, once we fix $\widehat{x}$ and $\widehat{y}$ from the ϵ-net, we can form the $\Re^{k\times n}$ matrix that directly measures the contribution by the entries in g^S.

$$W^T \mathrm{DIAG}\left(\widehat{y}\right) U^T.$$

So our goal is to prove that a random subset of columns of this matrix picked independently with probability h/n per column has high numerical rank.

To do so, we will show that for a fixed $\widehat{y}$, any vector in the span of W is likely to have non-zero dot products with most columns of the form of $\mathrm{DIAG}\left(\widehat{y}\right) U_{i,:}^T$. This relies on proving a global lower bound on the density of null space vectors of G^S. So we will carry it out as a global argument in Section 5.2

Lemma 5.3. *Let U be an n-by-n orthornomal basis, $\mathcal{Y}$ be a family of length n vectors with magnitude at most n, at least $n-m$ entries with magnitude at least $\alpha_Y < n^{-4\log n}$ for some $m < n^{\frac{1}{4}}$, and let G^S be an $n \times d$ sparse Gaussian matrix with each entry set to $\mathcal{N}(0,1)$ with probability h/n for some h such that $dh > 20n\log n\log(1/\alpha)$ and $d > 80m$. Then with probability at least*

$$1 - n^{-9} - |\mathcal{Y}|\,\alpha_Y^{\frac{d}{10}},$$

the matrix G^S has the property that for any vector w orthogonal to the columns of G^S, and any vector $\widehat{y} \in \mathcal{Y}$, the vector $w^T \mathrm{DIAG}\left(\widehat{y}\right) U^T$ has at least $\frac{d}{80}$ entries with magnitude at least α_Y^3.

We then convert this per-entry bound to an overall bound on the minimum singular value of $W^T \mathrm{DIAG}\left(y\right) U_{:,S}^T$, where S is the subset of non-zeros picked in g^S. This is once again done by union bound over an ϵ-net. Here the granularity of the net is again dictated by the minimum dot product that we want to show. This value is in turn related to the entry-wise magnitude lower bound of the matrix that we can assume, which is α_Y^3.

Lemma 5.4. *Let Y be an $r \times n$ matrix with $r \leqslant n$ and per-entry magnitude at most n^{10} such that for any unit vector $z \in \Re^r$, the vector $z^T Y$ has at least t entries with magnitude at least $\alpha_Y < n^{-12}$. Then a random sample of the columns of Y with each column chosen independently with probability $\frac{h}{n}$ gives a subset S such that for all unit $z \in \Re^k$,*

$$\left\| z^T Y_{:,S} \right\|_2 \geqslant \frac{\alpha_Y}{2}$$

with probabitliy at least

$$1 - \alpha_Y^{-2r} \exp\left(-\frac{t \cdot h}{n}\right)$$

Proof. The magnitude upper bound of Y ensures for any z the rounded vector $\widehat{z}$ has

$$\left\| z^T Y_{:,S} - \widehat{z}^T Y_{:,S} \right\|_2 \leqslant \left\| z - \widehat{z} \right\|_2 \cdot \left\| Y_{:,S} \right\|_2 \leqslant \sqrt{n} \left\| z - \widehat{z} \right\|_\infty \cdot \sqrt{n} \left\| Y_{:,S} \right\| \leqslant n^{11} \left\| z - \widehat{z} \right\|_\infty$$

where the second last inequality follows from bounding the ℓ_2 norm by ℓ_∞ norms. So an ϵ-net over z with granularity

$$\epsilon_Z = \frac{\alpha_Y}{2n^{11}}.$$

ensures that for any z, there is some $\widehat{z}$ in the net such that

$$\left\| z^T Y_{:,S} - \widehat{z}^T Y_{:,S} \right\|_2 \leqslant \frac{\alpha_Y}{2}.$$

So it suffices to bound the probability that for all $\widehat{z}$, S contains at least one of the large entries in the vector $\widehat{z}^T Y$. As each of the t large entries is picked with probability h/n, the probability that none gets picked is

$$\left(1 - \frac{h}{n}\right)^t \leqslant \exp\left(-\frac{ht}{n}\right).$$

While on the other hand, because every entry of a unit vector z has value between -1 and 1, the rounding to granularity ϵ_Z gives at most

$$\frac{2}{\epsilon_Z} \leqslant 4n^{11} \alpha_Y^{-1} \leqslant \alpha_Y^{-2}$$

values per coordinate. Multiplied over the r coordinates of z then gives that the total size of the ϵ-net is at most α_Y^{-2r}. Taking a union bound over all these vectors then gives the overall bound. ☐

This ensures that the sub-matrix corresponding to the non-zero entries of g^S is well conditioned. We can then apply *restricted invertibility* [BT87] to obtain a subset of columns $J \subseteq S$ such that the covariance of those Gaussian entries is well conditioned. The formal statement of restricted invertibility that we will use is from [SS12].

Lemma 5.5. *Let Y be a $n \times r$ matrix such that Y^TY has minimum singular value at least ξ. Then there is a subset $J \subseteq [n]$ of size at least $r/2$ such that $Y_{J,:}$ has rank $|J|$, and minimum singluar value at least $\frac{\xi}{\sqrt{10n}}$, or equivalently:*

$$Y_{J,:}Y_{J,:}^T \succeq \frac{\xi^2}{10n} I.$$

Proof. The isotropic, or $Y^TY = I$ case of this statement is Theorem 2 from [SS12] instantiated with:

- v_i being the ith row of Y,
- $n \leftarrow s$, $m \leftarrow n$,
- $L = I$,
- $\epsilon = 1/2$.

For the more general case, consider the matrix

$$M \leftarrow Y^TY.$$

and in turn the matrix $YM^{-1/2}$. This matrix satisfies

$$\left(YM^{-1/2}\right)^T YM^{-1/2} = M^{-1/2}Y^TYM^{-1/2} = M^{-1/2}MM^{-1/2} = I.$$

So we can apply the special case of restricted invertibility (for the case where the Gram matrix is identity) mentioned above to get a row subset J such that:

$$\frac{1}{10n} \preceq Y_{J,:}M^{-1/2}\left(Y_{J,:}M^{-1/2}\right)^T = Y_{J,:}M^{-1}Y_{J,:}^T$$

On the other hand, the singular value bound on Y implies $M \succeq \xi^2 I$ and therefore $M^{1/2} \succeq \xi I$. So we get for any unit vector $u \in \Re^{|J|}$,

$$\left\|Y_{J,:}^T u\right\|_2 \geqslant \xi \left\|M^{-1/2}Y_{J,:}^T u\right\|_2 \geqslant \xi\sqrt{\frac{1}{10n}},$$

which when squared gives the desired bound on singular values. $\square$

Proof. (of Lemma 5.2) As adding to columns of W can only increase $\left\|W^Tz\right\|$, it suffices to consider the case of $r = 5m$. Consider taking an ϵ-net over all unit $x \in \Re^m$ with granularity

$$\epsilon_x \geqslant n^{-10}\alpha_V^3.$$

The size of this net is at most

$$\left(\frac{2}{n^{-10}\alpha_V^3}\right)^m \leqslant \alpha_V^{-4m}.$$

Let the associated set of $\widehat{y} = V\widehat{x}$ vectors be $\mathcal{Y}$.

Then Lemma 5.3 gives that with probability at least

$$1 - n^{-9} - \alpha_V^{-4m} \alpha_V^{\frac{d}{10}} \geqslant 1 - n^{-8}.$$

the W matrix orthogonal to G^S has the property that for all w in its column span, and for all $\widehat{y} \in \mathcal{Y}$,

$$w^T \mathrm{DIAG}\left(\widehat{y}\right) U^T$$

has at least

$$\frac{s}{80} \geqslant \frac{n}{160m}$$

entries with magnitude at least α_V^2, Here the last inequality in the probability bound follows from the assumption of $m < n^{1/4}$.

Consider each individual $\widehat{y}$. Let $Y(\widehat{y})$ be the associated matrix

$$Y\left(\widehat{y}\right) = W^T \mathrm{DIAG}\left(\widehat{y}\right) U^T$$

Lemma 5.4 gives that the (globally) picked subset S give that $W^T \mathrm{DIAG}\left(\widehat{y}\right) U^T_{:,S} = Y(\widehat{y})_{:,S}$ has minimum singular value at least $\alpha_V^2/2$ with probability at least

$$1 - \left(2\alpha_V\right)^{-4m} \exp\left(-\frac{h}{n} \cdot \frac{n}{160m}\right) = 1 - \alpha_V^{-4m} \exp\left(-\frac{h}{200m}\right).$$

Should such bound hold, by restricted invertibility as stated in Lemma 5.5, we get that there is a set $J(\widehat{y})$ with size at least $r/2$ such that

$$Y\left(\widehat{y}\right)_{:,J(\widehat{y})}^T Y\left(\widehat{y}\right)_{:,J(\widehat{y})} \succeq \frac{\alpha_V^2}{10n} I.$$

This means it suffices to bound the probability over the random choice of g_S^S which lead to $\left\|Y(\widehat{y}) g_S^S\right\|_2$ being small. Specifically, we can do a worst-case bound over entries not in S, call that vector b, to get:

$$\Pr_{G_S^S}\left[\left\|Y\left(\widehat{y}\right)_{:,S} g_S^S\right\|_2 \leqslant \alpha_V^4\right] \leqslant \max_b \Pr_{G_{J(\widehat{y})}^S}\left[\left\|Y\left(\widehat{y}\right)_{:,J(\widehat{y})} g_{J(\widehat{y})}^S - b\right\|_2 \leqslant \alpha_V^4\right].$$

multiplying by $Y(\widehat{y})_{:,J(\widehat{y})}^T$ and $(Y(\widehat{y})_{:,J(\widehat{y})}^T Y(\widehat{y})_{:,J(\widehat{y})})^{-1}$ then gives

$$\begin{aligned}\left\|Y\left(\widehat{y}\right)_{:,J(\widehat{y})} g_{J(\widehat{y})}^S - b\right\|_2 &\geqslant n^{-10}\left\|Y\left(\widehat{y}\right)_{:,J(\widehat{y})}^T Y\left(\widehat{y}\right)_{:,J(\widehat{y})} g_{J(\widehat{y})}^S - Y\left(\widehat{y}\right)_{:,J(\widehat{y})} b\right\|_2 \\ &\geqslant \alpha_V^3 \left\|g_{J(\widehat{y})}^S - \left(Y\left(\widehat{y}\right)_{:,J(\widehat{y})}^T Y\left(\widehat{y}\right)_{:,J(\widehat{y})}\right)^{-1} Y\left(\widehat{y}\right)_{:,J(\widehat{y})} b\right\|_2.\end{aligned}$$

Here the first inequality follows from the magnitude upper bound on $Y(\widehat{y})_{:,J(\widehat{y})}^T$, and the bound on min singular value above.

Substituting this lower bound back in, with a different choice of the worst-case vector, gives

$$\Pr_{g_S^S}\left[\left\|Y\left(\widehat{y}\right)_{:,S}g_S^S\right\|_2 \leqslant \alpha_V^4\right] \leqslant \max_{\widehat{b}} \Pr_{g^S_{J(\widehat{y})}}\left[\left\|g^S_{J(\widehat{y})} - \widehat{b}\right\|_2 \leqslant \alpha_V^{-3}\cdot\alpha_V^4\right]$$

Simplifying to coordinates gives

$$\leqslant \prod_{j\in J(\widehat{y})} \max_{\widehat{b}_j} \Pr_{g_j^S}\left[\left|g_j^S - \widehat{b}_j\right| \leqslant \alpha_V\right] \leqslant \alpha_V^{\frac{r}{2}}.$$

Here the last inequality follows from the density of a standard Gaussian being at most 1.

So the overall failure probability is, by union bound, at most

$$\alpha_V^{-4m}\cdot\alpha_V^r + \alpha_V^{-5r}\exp\left(-\frac{h}{200m}\right),$$

where the first term is from union bounding over the entire net of the above probability, and the second term follows from the invocation of Lemma 5.4.

Substituting in $r = 5m$ gives that the first term is at most $\alpha_V^m \leqslant n^{-20}$. For the second term, incorporating $h \geqslant 10000m^2\log(1/\alpha_V)$ gives

$$\alpha_V^{-5r}\exp\left(-\frac{h}{200m}\right) \leqslant \alpha_V^{-25m}\cdot\alpha_V^{50m} \leqslant n^{-20}.$$

Summing these then gives the overall failure probability. □

5.2 Density Lower Bound on Vectors in Null Space

It remains to rule out all vectors such that $w\mathrm{DIAG}\left(\widehat{y}\right)U^T$ is sparse for some $\widehat{y}$ corresponding to some $\widehat{x}$ in the ϵ-net. We do so by leveraging the initial Krylov block G^S. We show that any particular vector is highly unlikely to be orthogonal to all s columns of G^S. For a dense n-by-s Gaussian G, the probability of any particular vector being (nearly) orthogonal to all columns is about $\exp(-O(s))$ due to all columns being independent. We will show that sparse Gaussians also produce a similar behavior. The proof then utilizes this per-vector bound together with an ϵ-net argument on vectors that are sparse in their representation under the basis U, with an additional case to take the few small entries of $\widehat{y}$ into account.

Lemma 5.3. *Let U be an n-by-n orthornomal basis, $\mathcal{Y}$ be a family of length n vectors with magnitude at most n, at least $n-m$ entries with magnitude at least $\alpha_Y < n^{-4\log n}$ for some $m < n^{\frac{1}{4}}$, and let G^S be an $n\times d$ sparse Gaussian matrix with each entry set to $\mathcal{N}(0,1)$ with probability h/n for some h such that $dh > 20n\log n\log(1/\alpha)$ and $d > 80m$. Then with probability at least*

$$1 - n^{-9} - |\mathcal{Y}|\,\alpha_Y^{\frac{d}{10}},$$

the matrix G^S has the property that for any vector w orthogonal to the columns of G^S, and any vector $\widehat{y}\in\mathcal{Y}$, the vector $w^T\mathrm{DIAG}\left(\widehat{y}\right)U^T$ has at least $\frac{d}{80}$ entries with magnitude at least α_Y^3.

We first show that G^S rules out any particular vector with reasonably large probability. This is again done in two steps: first by ruling out all sparse vectors using a successive ϵ-net argument similar to the proof of Lemma 4.3. This proof is slightly simplified due to all entries of G^S being completely independent, instead of correlated across the diagonal as in Lemma 4.3.

Lemma 5.6. *Let G^S be an $n \times d$ sparse Gaussian matrix with each entry set to $\mathcal{N}(0,1)$ with probability h/n, and 0 otherwise. If $hd \geqslant 20n\log^2 n$, then with probability at least $1 - n^{-10}$, all unit vectors w satisfying $(G^S)^T w = 0$ (i.e., orthogonal to all columns G^S) has at least $\frac{d}{40\log n}$ entries with magnitude at least $n^{-4\log n}$.*

Proof. By Claim 3.3, we may assume that all entries in G^S have magnitude at most n.

We will prove by induction for $i = 0 \ldots \lfloor \log_2(\frac{d}{40\log n}) \rfloor$ that with probability at least $1 - in^{-11}$, all unit vectors in the null space of G^S has at least 2^i entries with magnitude at least $n^{-4(i+1)}$.

The base case of $i = 0$ follows from a length n unit vector having an entry of magnitude at least $n^{-1/2}$. For the inductive case, we will build an ϵ-net with granularity

$$\epsilon_i \leftarrow n^{-4(i+1)}.$$

That is, we only consider vectors whose entries are integer multiples of ϵ_i. For a generic vector w, we can round each entry of it toward 0 to form $\widehat{w}$ such that:

- $\left\|\widehat{w}\right\|_2 \leqslant 1$.
- $(G^S)^T\widehat{w}$ is entry-wise small:

$$\left\|(G^S)^T\widehat{w}\right\|_\infty \leqslant n\left\|G^S\right\|_\infty \left\|w - \widehat{w}\right\|_\infty \leqslant n^2\epsilon_i.$$

So it suffices to show that the probability of G^S having an entry-wise small product with any $\widehat{w}$ with at most 2^i non-zeros is small. This is because any entry with magnitude less than ϵ_i will get rounded to 0. Furthermore, by the inductive hypothesis, it suffices to consider only w with at least 2^{i-1} entries with magnitude at least ϵ_{i-1}. Each such entry, when perturbed by ϵ_i, has magnitude at least $\epsilon_{i-1} - \epsilon_i \geqslant \epsilon_{i-1}/2$.

We will do so by union bound over all such vectors $\widehat{w}$. Because the columns of the probability that any column of G^S picks none of t entries is at most

$$\left(1 - \frac{h}{n}\right)^{2^{i-1}} \leqslant \exp\left(-\frac{h \cdot 2^{i-1}}{n}\right).$$

Furthermore, if one of these entries are picked, the resulting Gaussian corresponding to the product of that column of G^S against $\widehat{w}$ has variance at least $\epsilon_{i-1}/2$. Which means that it's in an interval of size at most $n^3\epsilon_i$ with probability at most

$$\frac{n^2\epsilon_i}{\epsilon_{i-1}/2} \leqslant n^{-1},$$

where the inequality follows from the choice of $\epsilon_i = n^{-4}\epsilon_{i-1}$. Taking union bound over these two events gives that the probability of a column of G^S having small dot product against $\widehat{w}$ is at most

$$\exp\left(-\frac{h2^{i-1}}{n}\right) + n^{-1} \leqslant 2\exp\left(-\min\left\{\frac{h \cdot 2^{i-1}}{n}, \log n\right\}\right)$$

which compounded over the d columns gives, and substituting in the assumption of $h \cdot d \geqslant 20n\log^2 n$ gives an overall probability of at most

$$2\exp\left(-\min\left\{\frac{d \cdot h \cdot 2^{i-1}}{n}, d\log n\right\}\right) \leqslant 2\exp\left(-\min\left\{10\log^2 n \cdot 2^i, d\log n\right\}\right).$$

On the other hand, the number of vectors with 2^i non-zeros, norm at most 2, and entries rounded to integer multiples of ϵ_i is at most

$$\binom{n}{2^i} \cdot (4/\epsilon_i)^{2^i} \leqslant (4n/\epsilon_i)^{2i} \leqslant \exp\left(5 \cdot (i+1) \cdot \ln n \cdot 2^i\right) \leqslant \exp\left(5\log^2 n \cdot 2^i\right).$$

Matching this against the two terms means we need:

- For any $i \geqslant 0$, we have $5\log^2 n2^i \leqslant 5\log^2 n2^i$, and this term is minimized when $i = 1$. So the first term is at most $2\exp(-5\log^2 n) \leqslant n^{-10}$.
- For the second term to be small, substituting in $2^i \leqslant \frac{d}{40\log n}$ gives

  $$\exp\left(5\log^2 n \cdot 2^i\right) \cdot 2\exp\left(-d\log n\right) \leqslant 2\exp\left(\frac{d\log n}{8} - d\log n\right) \leqslant 2\exp\left(-\frac{7d\log n}{8}\right),$$

  which is at most n^{-10} when d is larger than some absolute constant.

□

This means under a global event that happens with probability at least $1 - n^{-10}$, we only need to consider dense vectors in the column space of W. For each such vector, we can use its density to show that it's highly unlikely to have small product against G^S.

Lemma 5.7. *Let $0 < \alpha < n^{-8\log n}$ be a threshold, and let G^S be a $n \times d$ sparse Gaussian matrix with each entry set to $\mathcal{N}(0,1)$ with probability h/n for some $hd > 20n\log n\log(1/\alpha)$. Then any unit vector w with at least $\frac{d}{40\log n}$ entries with magnitude at least $n^{-4\log n}$ satisfies*

$$Pr_{G^S}\left[\left\|\left(G^S\right)^T w\right\|_2 < \alpha\right] < \alpha^{\frac{d}{5}}.$$

Proof. Consider each column of G^S, g^S. The probability that g^S picks none of the large entries in w is at most

$$\left(1 - \frac{h}{n}\right)^{\frac{d}{40\log n}} = \exp\left(-\frac{hd}{n \cdot 40\log n}\right) \leqslant \exp\left(-\frac{20n\log n \log\left(1/\alpha\right)}{40n\log n}\right) = \alpha^{\frac{1}{2}}.$$

In the case such an entry is picked, the resulting product with the Gaussian has variance at least $n^{-4\log n}$. So is in an interval of size α with probability at most

$$\frac{\alpha}{n^{-4\log n}} \leqslant \alpha^{\frac{1}{2}},$$

where the last inequality follows from the assumption of $\alpha < n^{-8\log n}$. By union bound, the probability of g^S having small dot product against w is at most $\alpha^{1/5}$. Compounding this over the d columns then gives the overall bound. □

The rest of the proof is an ϵ-net based argument on all w for which $w^T\text{DIAG}\left(\widehat{y}\right)U$ is sparse. For each $\widehat{y}$, we want to generate some set of vectors $\mathcal{W}(\widehat{y})$ such that if w is a vector where $w^T\text{DIAG}\left(\widehat{y}\right)U^T$ is sparse, there is some $\widehat{w} \in \mathcal{W}$ such that

$$\left\|\widehat{w} - w\right\|_2 \leqslant \alpha_Y.$$

After that, it suffices to show that all $\widehat{w} \in \mathcal{W}(\widehat{y})$ has large $\left\|\widehat{w}^T G^S\right\|_2$ via Lemma 5.7. Since U is invertible, most of w is recoverable from the $w^T\text{DIAG}\left(\widehat{y}\right)U$ vector via the operation

$$\left(w^T\text{DIAG}\left(\widehat{y}\right)U\right)U^T = w^T\text{DIAG}\left(\widehat{y}\right).$$

So up to a small number of coordinates corresponding to the small magnitude entries in $\widehat{y}$, we can enumerate over the possible $\widehat{w}$s by enumerating over the possible sparse w^TU vectors. For the non-zero coordinates in both the original and spectral domains, it also suffices to consider entries that are integer multiples of poly(α_Y).

Proof. (of Lemma 5.3) Claim 3.3 allows us to assume that the maximum magnitude in G^S is n.

We will use an ϵ-net argument, with the goal of invoking Lemma 5.7 on all vectors in the net. In order to do so, we first invoke Lemma 5.6, and pay for the global failure probability of n^{-10} once for all vectors $\widehat{y}$.

We also generate the set of vectors that are t-sparse in the U basis representation, with granularity

$$\epsilon_q \leftarrow \alpha_Y^3.$$

Formally, let $\mathcal{Q}$ denote all the vectors $\widehat{q}$ with norm at most 2 and at most t non-zeros, all of which are integer multiples of ϵ_q.

Now consider some vector $\widehat{y} \in \mathcal{Y}$. Let BIG be the subset of entries in y that are at least α_Y, and $\overline{BIG}$ its complement. We generate $\mathcal{W}(\widehat{y})$ by considering all vectors of the form

$$\begin{aligned}\widehat{w}_{BIG} &= \text{DIAG}\left(\widehat{y}_{BIG}\right)^{-1} U^T \widehat{q} \qquad \text{for some } \widehat{q} \in \mathcal{Q}\\ \widehat{w}_{\overline{BIG}} &= \text{integer multiples of } \alpha_Y^2\end{aligned}$$

Then we want to show that any w for which $w^T \text{DIAG}\left(\widehat{y}\right) U^T$ is t-sparse is close to some $\widehat{w} \in \mathcal{W}(\widehat{y})$. For such a w, consider the vector

$$q = U \text{DIAG}\left(\widehat{y}\right) w,$$

and suppose we rounded its entries to the nearest multiples of ϵ_q for some ϵ_q, giving $\widehat{q}$ such that $\left\|\widehat{q} - q\right\|_\infty \leqslant \epsilon_q$. Because U is an orthonormal matrix, any error in q translates to an error in $U^T q - \text{DIAG}\left(\widehat{y}\right) w$ as well:

$$\left\|U^T \widehat{q} - \text{DIAG}\left(\widehat{y}\right) w\right\|_2 \leqslant \left\|\widehat{q} - U \text{DIAG}\left(\widehat{y}\right) w\right\|_2 = \left\|\widehat{q} - q\right\|_2 \leqslant n^{1/2} \epsilon_q.$$

This error can in turn be carried across all entries in $\widehat{y}_{BIG}$, using the entry-wise lower bounds. Specifcally, for all $i \in BIG$ we have

$$\left|\left(U^T \widehat{q}\right)_i \widehat{y}_i^{-1} - w_i\right| \leqslant \left|\widehat{y}_i^{-1}\right| \left|\left(U^T \widehat{q}\right)_i - \widehat{y}_i w_i\right| \leqslant \alpha_Y^{-1} \cdot \left\|U^T \widehat{q} - \text{DIAG}\left(\widehat{y}\right) w\right\|_2 \leqslant \alpha_Y^{-1} n^{1/2} \epsilon_q.$$

Thus the $\widehat{w}$ that corresponds to this w is the one from this $\widehat{q}$, plus having all entries in $\overline{BIG}$ rounded explicitly. That is, for any such w with $w^T \text{DIAG}\left(\widehat{y}\right) U^T$ t-sparse, there is some $\widehat{w} \in \mathcal{W}(\widehat{y})$ such that

$$\left\|w - \widehat{w}\right\|_2 \leqslant n^2 \alpha_Y^{-1} \epsilon_q \leqslant \frac{\alpha_Y}{n^3},$$

where the inequality follows from the choice of ϵ_q and the assumption of $\alpha_Y \leqslant n^{-8 \log n}$. Combining with the assumption of max magnitude in G^S being n from Claim 3.3 also gives

$$\left\|G^S w - G^S \widehat{w}\right\|_2 \leqslant \frac{\alpha_Y}{2}.$$

Thus, to rule out all such w, it suffices to show that all $\widehat{w} \in \mathcal{W}(\widehat{y})$ have $\left\|(G^S)^T \widehat{w}\right\|_2 \geqslant \alpha_Y$. We do so by taking union bound over the entire ϵ-net.

Since w is a unit vector, $U^T w$ also has norm at most 1. Combining this with the assumption of the entries of $\widehat{y}$ having magnitude at most n gives that the max magnitude of an entry in $\widehat{q}$ is at most $2n$. As each of the t non-zeros in $\overline{BIG}$ is explicitly enumerated with magnitude at most $\epsilon_q = \alpha_Y^3$, we have:

$$|\mathcal{Q}| \leqslant \binom{n}{t} \cdot \left(4n\alpha_Y^{-3}\right)^t \leqslant \left(4n^2\alpha_Y^{-3}\right)^t \leqslant \alpha_Y^{-4t}$$

which combined with the m entries of $\widehat{w}$ being explicitly generated as multiples of ϵ_q gives

$$\left|\mathcal{W}\left(\widehat{y}\right)\right| \leqslant \alpha_Y^{-4(t+m)}.$$

We remark that the key in this step is that the overhead from generating terms in $\widehat{w}$ has m in the exponent instead of n.

As we've already globally conditioned on all vectors in the null space of G^S being dense, we get that each vector $\widehat{w} \in \mathcal{W}$ gives a small dot product with probability at most $\alpha_Y^{-\frac{d}{5}}$. Taking union bound over all $|\mathcal{Y}| \cdot \alpha_Y^{-4(t+m)}$ vectors then gives a failure probability of at most

$$|\mathcal{Y}| \cdot \alpha_Y^{-4(t+m)+\frac{d}{5}}.$$

When $m, t \leqslant \frac{d}{80}$, this is at most

$$|\mathcal{Y}| \cdot \alpha_Y^{-\frac{d}{10}+\frac{d}{5}} = |\mathcal{Y}| \cdot \alpha_Y^{\frac{d}{10}},$$

which is the desired bound. □

Note that the need to rule out all sparse vectors in the null space precludes us from applying this bound separately for each vector $\widehat{y}$ in the ϵ-net. Instead, we lower bound the density of all null space vectors via Lemma 5.6 once for all $\widehat{y}$ in the ϵ-net, and then invoke Lemma 5.7 for each of the nets generated for each $\widehat{y}$.

6 Solver for Matrices with Low Displacement Rank

We now prove the running time of the solver for Hankel matrices. Our notation of matrices will revolve around block matrices throughout that section: we use s to denote the size of a block, and m to denote the number of blocks. When there are multiple matrices that form natural sequences, we will index into them using superscripts.

The algorithm here has much similarities with the hierarchical matrix based solver by Xia, Xi, and Gu [XXG12]. The main difference is that we work in the matrix domain instead of the Fourier domain, and our algorithm is optimized for inputs with arbitrarily lengthed simulated floats. Xia, Xi, and Gu [XXG12] does most of what we do: after transforming the problem to the Fourier domain, they write the matrix as a Cauchy matrix, which they in turn view as a hierarchical matrix multiplied by complex coefficients of the form of $\frac{1}{z^i - z^j}$ (z is a complex root of unity). By leveraging well-spaced decompositions similar to the fast multipole method, they are able to use stable solvers for hierarchical matrices to extract the solution to the overall Cauchy matrix. To our knowledge, directly invoking this algorithm would lead to an extra factor of m. This is because our condition number, and sizes of the numbers involved, are all $\exp(\widetilde{O}(m))$. The fast multipole method is only able to extra one digit per iteration due to its reliance on the Taylor expansion, so would give a total running time of $s^{\omega} m^3 > n^{\omega}$, which is too big.

We will use the $\{\cdot\}$ notation to index into subsets of blocks, in the same manner as indexing into row/column indices.

Definition 6.1. Given block size s and a set of indices $S \subseteq [m]$, we use $\{S\}$ to denote the entries in the corresponding blocks. That is, if we arrange the indices of the m blocks sequentially, we have:

$$\{S\} = \bigcup_{i \in S} \left[(i-1)\,s + 1, is\right].$$

Theorem 3.8. *If H is an $sm \times sm$ symmetric s-block-Hankel matrix and $0 < \alpha_H < (sm)^{-100}$ is a parameter such that every contiguous square block-aligned minor of H containing the top-right or bottom left corner have minimum eigenvalue at least α_H:*

$$\sigma_{\min}\left(H_{\{1:i,(m-i+1):m\}}\right), \sigma_{\min}\left(H_{\{(m-i+1):m,1:i\}}\right) \geqslant \alpha_H \qquad \forall 1 \leqslant i \leqslant m$$

and all entries in H have magnitude at most $(sm)^{-2}\alpha_H^{-1}$, then for any error ϵ, we can preprocess H in time $\widetilde{O}(ms^{\omega}\log(\alpha_H^{-1}\epsilon^{-1}))$ to form $\textsc{Solve}_H(\cdot, \epsilon)$ that corresponds to a linear operator Z_H such that:

1. *For any $(ms) \times k$ matrix B with max magnitude $\|\!|B|\!\|_\infty$ and L_B words after the decimal point, $\textsc{Solve}_H(B, \epsilon)$ returns $Z_H B$ in time*

$$\widetilde{O}\left(m \cdot \max\left\{s^{\omega-1}k, s^2 k^{\omega-2}\right\} \cdot \left(\log\left(\frac{(1+\|\!|B|\!\|_\infty)\,ms}{\alpha_H \epsilon}\right) + L_B\right)\right).$$

2. *Z_H is a high-accuracy approximation to H^{-1}:*

$$\left\|Z_H - H^{-1}\right\|_F \leqslant \epsilon.$$

3. *the entries of Z_H have at most $O(\log^2 m \log(\alpha_H^{-1}\epsilon^{-1}))$ words after the decimal point.*

Crucial to our analysis is the small displacement rank property of the Hankel matrix. This fact relies on the s-block-Hankel matrix is identical to itself shifted down and to the left by s entries each. This down/left shift however is a bit more cumbersome to represent notationally, as the shifts occur in different directions along the rows and columns. So instead, we work with s-block-Toeplitz matrices, which is formed by reversing the order of columns of H.

Definition 6.2. A s-block-Toeplitz matrix with m blocks is an ms-by-ms matrix T where

$$T_{\{i,j\}} = M^{(i-j)}$$

where $M^{(-m+1)} \ldots M^{(m-1)}$ is a sequence of s-by-s matrices.

Notationally we will use T to denote the matrices that we operate on to emphasize the connection/motivation with s-block-Toeplitz matrices.

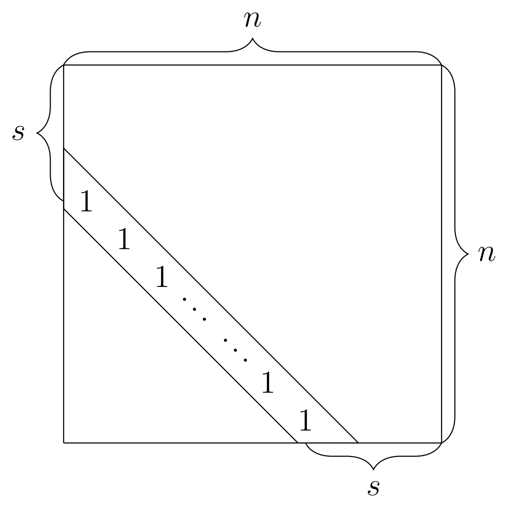

Figure 6: Displacement marix $\Delta(s) \in \Re^{n\times n}$

6.1 Displacement Rank Based Representations

We can then define the shift-down by s operator. Its transpose is the shift-right by s operator when right multiplied to the matrices. We will fix this definition for our choice of block size of s. An illustration of it is in Figure 6.

Definition 6.3. For any choice of block size s and block number m, the square displacement operator $\Delta(s)$ (whose dimension we assume to be implicit to the matrix we use it against) is the matrix with 1s on all entries s below the diagonal, and 0 everywhere else.

$$\Delta_{ij} = \begin{cases} 1\,(s) & \text{if } i = j + s \\ 0 & \text{otherwise.} \end{cases}$$

Then for an n-by-n matrix M, the s^+/s^--displaced versions of M are given by:

$$\begin{aligned} s^+(M) &= M - \Delta(s)\,M\Delta(s)^T, \\ s^-(M) &= M - \Delta(s)^T M\Delta(s). \end{aligned}$$

and its $+s/-s$-displaced ranks are:

$$\begin{aligned} \textsc{Rank}_{+s}(M) &= \textsc{Rank}\left(s^+(M)\right) = \textsc{Rank}\left(M - \Delta(s)\,M\Delta(s)^T\right), \\ \textsc{Rank}_{-s}(M) &= \textsc{Rank}\left(s^-(M)\right) = \textsc{Rank}\left(M - \Delta(s)^T M\Delta(s)\right). \end{aligned}$$

Observe that if T is a s-block-Toeplitz matrix, then every leading principle minor of T has s^+-displacement rank at most $2s$. The key property of displacement matrices is that the inverse of a full rank matrix has the same displacement rank under a sign flip. The following is an adapation of Theorem 1 from [KKM79] to the more general displacement setting.

Lemma 6.4. *For any invertible matrix M and any shift value s, we have*

$$\textsc{Rank}_{+s}\left(M\right) = \textsc{Rank}_{-s}\left(M^{-1}\right)$$

Note that this Lemma applied with M^{-1} instead of M also gives $\textsc{Rank}_{-s}(M) = \textsc{Rank}_{+s}(M^{-1})$. The $s = 1$ case of this is also the reason behind the representation of inverses of Toeplitz matrices known as the Gohberg-Krupnik Formula [LS92, GK72].

Lemma 6.4 allows us to have rank-s representations of inverses of leading minors of T, as well as the Schur complements formed when inverting onto a subset of the entries.

Also observe that matrix-vector multiplications involving $\Delta(s)$ and $\Delta(s)^T$ take linear time: it's merely shifting all entries. So given multiplication access to M, we can also obtain multiplication access to both $s^+(M)$ and $s^-(M)$. Such a translation in representations is also error-preserving. We check that errors in the orignal matrix, or the displaced versions, translate naturally to each toher.

Lemma 6.5. *For any n-by-n matrices M and $\widetilde{M}$*

$$n^{-2}\left\|M - \widetilde{M}\right\|_F \leqslant \left\|s^+\left(M\right) - s^+\left(\widetilde{M}\right)\right\|_F \leqslant n^2\left\|M - \widetilde{M}\right\|_F$$

and similarly for the differences of the negatively displaced versions.

Proof. In the forward direction, we have for all $i, j \geqslant s$,

$$\begin{aligned}\left|s^+\left(M\right)_{ij} - s^+\left(\widetilde{M}\right)_{ij}\right| &= \left|M_{ij} - M_{i-s,j-s} - \left(\widetilde{M}_{ij} - \widetilde{M}_{i-s,j-s}\right)\right| \\ &\leqslant \left|M_{ij} - \widetilde{M}_{ij}\right| + \left|M_{i-s,j-s} - \widetilde{M}_{i-s,j-s}\right|\end{aligned}$$

and the rest of the entries are the same. So each entry in the difference $M - \widetilde{M}$ contributes to at most two entries.

In the reverse direction, we get

$$M_{ij} = \sum_{0 \leqslant k \leqslant \lfloor i/s \rfloor} s^+\left(M\right)_{i-ks,j-ks}$$

which subtracted against the same formula for $\widetilde{A}$ gives

$$\begin{aligned}\left|M_{ij} - \widetilde{M}_{ij}\right| &= \left|\sum_k s^+\left(M\right)_{i-ks,j-ks} - s^+\left(\widetilde{M}\right)_{i-ks,j-ks}\right| \\ &\leqslant \sum_k \left|s^+\left(M\right)_{i-ks,j-ks} - s^+\left(\widetilde{M}\right)_{i-ks,j-ks}\right|.\end{aligned}$$

So the contributions of errors on each entry get amplified by a factor of at most n. □

We will treat this representation as a black-box, and formalize interactions with it using the following lemma.

Lemma 6.6. *Given block size s, block count m, (ms)-by-r matrices X and Y, the (ms)-by-(ms) matrix M such that*

$$M - \Delta(s) M \Delta(s)^T = X^T Y$$

has a unique solution.

Furthermore, there is a routine ImplicitMatVec *that for any accuracy $\delta < (ms)^{-10}$ corresponds to a linear operator $\widetilde{Z}_{XY \to M,\delta}$ such that for any ms-by-k matrix B with at most L_B words after decimal place,* ImplicitMatVec(X, Y, B, δ) *takes time (measured in number of word operations)*

$$O\left(m \log^3 m \cdot \max\{r, s\} \max\left\{s^{\omega-2}k, sk^{\omega-2}\right\} \cdot \left(L_B + \log\left(\left(1 + \|\!|X|\!\|_\infty\right)\left(1 + \|\!|Y|\!\|_\infty\right)\left(1 + \|\!|B|\!\|_\infty\right) ms/\delta\right)\right)\right)$$

and outputs the (ms)-by-k matrix

$$\widetilde{Z}_{XY \to M,\delta} B$$

where $\widetilde{Z}_{XY \to M,\delta}$ is a matrix with at most $O(\log m \log((1+\|\!|X|\!\|_\infty)(1+\|\!|Y|\!\|_\infty)ms/\delta))$ words after the decimal point such that

$$\left\| \widetilde{Z}_{XY \to M,\delta} - M \right\|_F \leqslant \delta$$

The same guarantees and runtime bounds also hold for the negative displacement case where M is implicitly specified as $M - \Delta(s)^T M \Delta(s) = XY^T$, as well as matrix-vector multiplications with M^T.

6.2 Recursive Schur Complement

The main algorithm is to invoke this succinct representation during intermediate steps of (block) Gaussian elimination. Specifically, for a subset of coordinates of C and its complement set $\overline{C}$, we want to directly produce (a high accuracy approximation of) the low displacement rank factorization of

$$\mathrm{SC}(T, C) = T_{CC} - T_{C\overline{C}} T_{\overline{C}\overline{C}}^{-1} T_{\overline{C}C}.$$

Before proceeding with the algorithm we first must show that the Schur complement has small displacement rank. For this, we need the following characterization of Schur complements as minors of inverses.

Fact 6.7. *If M is a full rank matrix, then for any subset of coordinates C, we have*

$$\mathrm{SC}(M, C)^{-1} = \left[M^{-1}\right]_{CC}.$$

Combining this with the fact that positive/negative displacement ranks work well under taking leading/trailing principle minors gives the following bounds on displacement ranks. We will in general use C to denote the remaining coordinates, which in our recursive algorithm will be a suffix of the indices. Then the leading portion of coordinates will be denoted using $\overline{C}$.

Lemma 6.8. *If M is a symmetric full rank matrix, C and $\overline{C}$ are a coordinate wise suffix/prefix split of the indices, then we have:*

$$\textsc{Rank}_{+s}\left(M_{\overline{C},\overline{C}}\right), \textsc{Rank}_{+s}\left(\mathrm{SC}\left(M,C\right)\right) \leqslant \textsc{Rank}_{+s}\left(M\right)$$

Proof. To bound the displacement rank of the leading minor $M_{\overline{C}\overline{C}}$, note that because $\overline{C}$ is a prefix of the coordinates,

$$\Delta\left(s\right) M_{\overline{C}\overline{C}} \Delta\left(s\right)^T = \left(M - \Delta\left(s\right) M \Delta\left(s\right)^T\right)_{\overline{C}\overline{C}}$$

which when substituted back in gives

$$M_{\overline{C}\overline{C}} - \Delta\left(s\right) M_{\overline{C}\overline{C}} \Delta\left(s\right)^T = \left(M - \Delta\left(s\right) M \Delta\left(s\right)^T\right)_{\overline{C}\overline{C}}.$$

The inequality then follows from minors of matrices having smaller ranks.

For the Schur Complement onto the suffix of indices C, we combine the inverse representation of Schur Complements from Fact 6.7 above with the relation between positive and negative displacement ranks from Lemma 6.4.

Because C is a suffix of the indices, we get

$$M_{CC}^{-1} - \Delta\left(s\right)^T M_{CC}^{-1} \Delta\left(s\right) = \left(M^{-1} - \Delta\left(s\right) M^{-1} \Delta\left(s\right)^T\right)_{CC},$$

which implies

$$\textsc{Rank}_{+s}\left(\mathrm{SC}\left(M,C\right)\right) = \textsc{Rank}_{-s}\left(M_{CC}^{-1}\right) \leqslant \textsc{Rank}_{-s}\left(M^{-1}\right) = \textsc{Rank}_{+s}\left(M\right)$$

where the first and last equalities follow from the equivalences between positive and negative displacement ranks given by Lemma 6.4. □

This observation, plus the low displacement representation given in Lemma 6.6 leads to a recursive routine. We repeatedly partitioning up the matrix into two even halves of leading/trailing coordinates. As our partitions (as well as initial eigenvalue conditions) are on the blocks, we will overload notation and use $\{\overline{C}\}$ and $\{C\}$ to denote the splits into the corresponding (size s) blocks.

Then we recursively find the inverse of the top-left half $\{\overline{C}\}$ (in the succinct representation given by Lemma 6.6), and use it to compute the Schur complemt of the bottom-right half on $\{C\}$. Then we once can recurse again on the Schur complement on $\{C\}$, which also has low displacement rank representation. Combining its inverse with the inverse of $T_{\{\overline{C},\overline{C}\}}$ then gives the overall inverse.

Top-level pseudocode of this method is in Figure 7 Its main algorithmic difficulties lie in directly computing the displaced factorizations of the Schur complement, and the overall inverse. We present this algorithm in Section 6.3 before returning to the overall proof.

```
$\textsc{Solve}_T(\cdot) = \textsc{RecursiveSC}(m, s, X, Y, \epsilon)$
Input: $(ms) \times r$ matrices $X$ and $Y$ that describe an $(ms)$-by-$(ms)$ $s$-block-matrix $T$ with
$m \times m$ blocks that has $s^+$-displacement rank $r$. Error threshold $\epsilon$.
Output: A routine $\textsc{Solve}_T(\cdot)$ that corresponds to a linear operator $Z(T)$.

  1. If $m = 1$, return the explicit inverse of $T$.
  2. Let $\{C\}$ the last $\lfloor m/2 \rfloor$ blocks, and its complement $\{\overline{C}\}$ be the first $\lceil m/2 \rceil$ blocks.
  3. Use Lemma 6.6 to generate, using $X$ and $Y$, $\epsilon$-error multiplications involving $T_{C,C}$,
     $T_{\overline{C},C}$, and $T_{C,\overline{C}}$.
  4. Recursively on $\overline{C}$ to obtain operator $Z(\overline{C})$ that corresponds to
         $\textsc{Solve}_{T_{\{\overline{C},\overline{C}\}}}(\cdot) \leftarrow \textsc{RecursiveSC}\left(\lceil m/2 \rceil, s, X_{\{\overline{C}\},:}, Y_{\{\overline{C}\},:}, \epsilon\right).$
  5. Implicitly generate matrix multiplication functions for the Schur complement
         $\widetilde{SC} = T_{\{C,C\}} - T_{\{C,\overline{C}\}} Z(\overline{C}) T_{\{\overline{C},C\}},$
  6. Factorize the $s^+$-displaced approximate Schur Complement onto $\{C\}$:
         $X(\mathrm{SC}), Y(\mathrm{SC}) \leftarrow \textsc{LowRankApprox}(|C|, s, \textsc{Mult}_{s^+(\widetilde{SC})}(\cdot), \textsc{Mult}_{s^+(\widetilde{SC})^T}(\cdot), \epsilon).$
  7. Recurse on $\mathrm{SC}(T, C)$ to obtain operator $Z(SC)$ that corresponds to
         $\textsc{Solve}_{SC}(\cdot) \leftarrow \textsc{RecursiveSC}(\lfloor m/2 \rfloor, 2s, X(\mathrm{SC}), Y(\mathrm{SC}), \epsilon).$
  8. Use $Z(\overline{C})$ and $Z(SC)$ to implicitly generate multiplication operators for the $s^-$-
     displaced version of the approximate inverse operator
         $Z = \begin{bmatrix} I & -Z(\overline{C}) T_{\{\overline{C},C\}} \\ 0 & I \end{bmatrix} \begin{bmatrix} Z(\overline{C}) & 0 \\ 0 & Z(\mathrm{SC}) \end{bmatrix} \begin{bmatrix} I & 0 \\ -T_{\{C,\overline{C}\}} Z(F) & I \end{bmatrix},$
  9. Compute
         $X^{INV}, Y^{INV} \leftarrow \textsc{LowRankApprox}(m, 2s, \textsc{Mult}_{s^-(Z)}(\cdot), \textsc{Mult}_{s^-(Z)^T}(\cdot), \epsilon)$
     and return the multiplication operator given by Lemma 6.6 with error $\epsilon$.
```

Figure 7: Pseudocode for Recursive Schur Complement Algorithm

6.3 Implicit Low Rank Factorizations

Key to the efficiency of this algorithm is the ability to encode the $(sm)^2$ numbers of an inverse with ms^2 numbers instead. We show that given multiplication access to a matrix T. we can directly compute low rank factorizations of $s^+(T)$ and $s^-(T)$ by only calling the multiplication routine against (ms)-by-$O(s)$ sized matrices. For this section, we set $n = ms$, and $r = s$, so we work with n-by-n matrices that are close to rank r within some very small error.

The square case of computing rank revealing factoriaztions was studied in conjunction with the stability of fast matrix operations [DDH07]. However, we need to obtain running times sublinear in the matrix sizes.

We use random Gaussian projections, a method first analyzed by Sarlos [Sar06]. The quality of this random projection has been studied extensively in randomized numerical liner algebra. More recent progress show that a sparse random projection to about $\widetilde{O}(r)$ columns captures most of the column space information [Sar06, DMM08, KV17, Woo14]. However, because our matrix is given as black-box access, the density of the vectors do not affect the performance of our algorithm.

Specifically, we multiply the matrix to be factorized with a random Gaussian matrix with about r columns, and use that n-by-$\widetilde{O}(r)$ matrix to compute a good column basis for the entire matrix. The projection guarantees that we will use is the following variant of Theorem 14 from [Sar06], as well as Theorem 45 from [Woo14].

Lemma 6.9. *Let M be an $n \times n$ matrix, r any rank parameter, and $M(r)$ the best rank r approximation to M. Let S be a random n-by-$O(r)$ matrix with entries set i.i.d. to $N(0,1)$, and Π_{MS} the projection operator onto the column space of MS. Then we have with probability at least $1 - n^{-10}$,*

1. *The projection of M onto the row space of SM has small distance to $M(r)$:*

$$\left\|\Pi_{MS}M - M\left(r\right)\right\|_F \leqslant n^{30}\left\|M - M\left(r\right)\right\|_F.$$

2. *there is an #cols(S)-by-n matrix R with entry-wise magnitude at most $O(n^4)$ such that*

$$\left\|MSR - M\left(r\right)\right\|_F \leqslant n^{30}\left\|M - M\left(r\right)\right\|_F$$

Note that the success probability is set to $1 - n^{-10}$ instead of $1/2$ as in [Sar06]: this is at the expense of a larger error parameter. This modification is obtained by invoking the Markov inequality toward the end of the proof in [Sar06] with a larger error threshold.

To show the existence of R with small entry-wise magnitudes, we open up the proof of Lemma 45 of [Woo14], but transform it to work with matrices that reduce number of columns. In order to do so, we make use of the following two properties of the random Gaussian projections shown in [Woo14].

Lemma 6.10. *There is an absolute constant such that for any r, the n-by-$O(r)$ dense Gaussian matrix G satisfies:*

- *(Subspace Embedding Property) For any r-by-n matrix M, with probability at least $1 - n^{-20}$ we have*

$$0.9 \left\|x^T M\right\|_2 \leqslant \left\|x^T M S\right\|_2 \leqslant 1.1 \left\|x^T M\right\|_2 \qquad \forall x \in \Re^r$$

- *(Approximate Matrix Multiplication Property) For any two matrices $M(1)$ and $M(2)$ with n rows each, with probability at least $1 - n^{-20}$ we have*

$$\left\|M(1)^T S S^T M(2) - M(1)^T M(2)\right\|_F \leqslant n^{21} \left\|M(1)\right\|_F \left\|M(2)\right\|_F$$

Proof. (Of Lemma 6.9) Let the SVD of M be

$$M = U \Sigma V^T$$

and let the top r singular vectors/values be $U(r) \in \Re^{n \times s}$, $\Sigma(r) \in \Re^{r \times r}$, and $V(r) \in \Re^{n \times s}$ respectively. The given condition of the distance from M to its rank r approximation being at most ϵ means we have

$$\left\|M - U(r) \Sigma(r) V(r)^T\right\|_F \leqslant \epsilon$$

The matrix used to bound the distance is then

$$R = \left(V(r)^T S\right)^\dagger V(r)^T.$$

The subspace embedding property gives that because $V(r)$ has r rows, $V(r)^T S$ has full row rank r, and

$$\left(V(r)^T S\right)^\dagger = \left(V(r)^T S\right)^T Q^{-1} = S^T V(r) Q^{-1}$$

for some r-by-r matrix $Q = V(r)^T S S^T V(r)$ whose eigenvalues are in the range $[0.8, 1.3]$. This in turn means that the entry-wise magnitude of both Q and Q^{-1} are at most $O(1)$.

Combining this with the entry-wise magnitude of 1 for $V(r)$ (because it's an orthonormal basis) and n for S (due to Claim 3.3) gives that R has entry-wise magnitude at most $O(n^4)$, and Frobenius norm at most $O(n^5)$.

It remains to bound the error term. Let $\widehat{M}$ be the matrix corresponding to singular values $r+1 \ldots n$ in the SVD:

$$\widehat{M} = M - U(r) \Sigma(r) V(r)^T = U(n-r) \Sigma(n-r) V(n-r)^T.$$

The fact that Q perfectly inverts $V(r)^T S$ means $U(r)\Sigma(s)V(r)^T S R = U(r)\Sigma(s)V(r)$. So the error is only on the $\widehat{M}$ term:

$$M - MSR = \widehat{M} S R = \widehat{M} S S^T V(r) Q^{-1} V(r)^T.$$

Furthermore, the fact that $V(r)$ is an orthonormal basis, and that Q^{-1} has max eigenvalue 2, means that we can remove them from consideration:

$$\left\|\widehat{M}SR\right\|_F \leqslant \left\|\widehat{M}SS^TV\left(r\right)\right\|_F \left\|Q^{-1}\right\|_2 \left\|V\left(r\right)\right\|_2 \leqslant 2\left\|\widehat{M}SS^TV\left(r\right)\right\|_F.$$

For this last term, the orthgonality of singular vectors gives

$$0 = V\left(n-r\right)^T V\left(r\right) = U\left(n-r\right)\Sigma\left(n-r\right)V\left(n-r\right)^T V\left(r\right) = \widehat{M}V\left(r\right).$$

So applying the Approximate Matrix Multiplication Property gives

$$\left\|\widehat{M}SS^TV\left(r\right)\right\|_F \leqslant n^{21}\cdot\left\|\widehat{M}\right\|_F\cdot\left\|V\left(r\right)^T\right\|_F \leqslant n^{22}\epsilon.$$

Here the last inequality follows from $V(r)$ being an orthonormal basis. Incorporating the additional factor of 2 above then gives the result. □

We remark that this R matrix is used throughout randomized algorithms for computing low rank approximations [KV17]. The bound on the max magnitude of entries of R here allows us to perturb MS slightly so that its minimum singular value bounded away from 0, which in turn gives bounds on the bit-complexity of computing projections into its column space. Specifically, we invoke the dot-product against null space idea inherent to analyses of min singular values of entry-wise i.i.d. matrices [SST06, TV10], which is also the starting point of the proof of Theorem 3.7 in Section 5.

Lemma 6.11. *Let $\widehat{M}$ be any n-by-d matrix with $d < n$, and $\widetilde{M}$ a perturbation of $\widehat{M}$ formed by adding $\epsilon N(0,1)$ to every entry. Then with probability at least $1-n^{-11}$, the minimum singular value of $\widetilde{M}$ is at least ϵn^{-20}.*

Proof. For each column j, consider a vector $w(\backslash j)$ in the null space of the rest of the columns of $\widetilde{M}$. The dense Gaussian added to $\widetilde{M}$ gives that with probability at least $1-n^{-14}$, we have

$$\left|w\left(\backslash j\right)^T\widetilde{M}_{:j}\right| \geqslant \epsilon n^{-15}.$$

Taking union bound over all columnsn j gives that this holds for all columnns with probability at least $1-n^{-13}$.

We now lower bound $\left\|\widetilde{M}x\right\|_2$ for all unit vectors $x \in \Re^d$. For any such unit x, there is some j such that $|x_j| \geqslant n^{-1/2} > n^{-1}$. Then we get:

$$\left\|\widetilde{M}x\right\|_2 \geqslant |x_j|\cdot\left|w\left(\backslash j\right)^T\widetilde{M}_{:,j}\right| \geqslant \epsilon n^{-20}$$

where the last inequality follows from the dot-product lower bound above. □

Pseudocode of algorithm that utilizes this projection is in Figure 8. Its guarantees are given below in Lemma 6.12.

```
$(X, Y) = \textsc{LowRankApprox}(n, \textsc{Mult}_M(\cdot), \textsc{Mult}_{M^T}(\cdot), r, \epsilon)$
Input: implicit access to an $n \times n$ matrix $M$ via multiplication functions of it (via. $\textsc{Mult}_M(\cdot)$) and its transpose (via. $\textsc{Mult}_{M^T}(\cdot)$).
Target rank $r$ and error guarantee $\epsilon$.
Output: rank $r$ approximation of $M$ in factorized form, $X, Y \in \Re^{n \times r}$.

  1. Set $\widehat{\epsilon} \leftarrow \epsilon \left\lVert\!\left\lvert M \right\rvert\!\right\rVert_\infty^{-1} n^{-10}$
  2. Generate $n$-by-$O(r)$ random matrix $S$ with entries i.i.d. Gaussian, $N(0,1)$.
  3. Compute $\widehat{M} \leftarrow \textsc{Mult}_M(S, \widehat{\epsilon})$
  4. Perturb $\widehat{M}$ entry-wise by $\epsilon \cdot N(0,1)$ to form $\widetilde{M}$.
  5. Let $\widehat{X}$ be an orthonormal basis spanning the columns of $\widetilde{M}$,
         $\widehat{X} \leftarrow \widetilde{M} \left(\widetilde{M}^T \widetilde{M}\right)^{-1/2}$
     computed to additive accuracy $\widehat{\epsilon}$.
  6. Set $\widehat{Y} \leftarrow \textsc{Mult}_{M^T}(\widehat{X}, \widehat{\epsilon})$.
  7. Return the rank $r$ singular value decomposition to $\widehat{X}\,\widehat{Y}^T$.
```

Figure 8: Pseudocode for computing a rank r factorization of a matrix using only accesses to its products against vectors.

Lemma 6.12. *For an $n \times n$ matrix M with L_M words after the decimal point, and Frobenius norm distance at most ϵ to some rank r matrix, given via implicit access to multiplication operators $\textsc{Mult}_M$ and $\textsc{Mult}_{M^T}$, $\textsc{LowRankApprox}(n, \textsc{Mult}_M(\cdot), \textsc{Mult}_{M^T}(\cdot), r)$ returns with probability at least $1 - n^{-10}$, $X, Y \in \Re^{n \times r}$ such that*

1. *X and Y have magnitude at most $n^2 \left\lVert\!\left\lvert M \right\rvert\!\right\rVert_\infty$, and at most $\log((1+\left\lVert\!\left\lvert M \right\rvert\!\right\rVert_\infty)n/\epsilon)$ words after the decimal point,*
2. *The approximation error satisfies*

$$\left\lVert M - XY^T \right\rVert_F \leqslant n^{20}\epsilon$$

and the total cost of the algorithm is:

1. *the cost of calling $\textsc{Mult}_M(\cdot)$ and $\textsc{Mult}_{M^T}(\cdot)$ for $O(r \log n)$ vectors with entry magnitude at most 1 and at most $O(\log((1+\left\lVert\!\left\lvert M \right\rvert\!\right\rVert_\infty)n/\epsilon))$ words after the decimal point, plus*

2. an additional overhead of $\widetilde{O}(nr^{\omega-1}(\log((1+\|\|M\|\|_{\infty})n/\epsilon))$.

Proof. Lemma 6.9 gives that there is some R with entry-wise magnitude at most $O(n^4)$ such that

$$\left\|\widehat{M}R - M\right\|_F \leqslant n^{30}\epsilon.$$

Furthermore, the entry-wise perturbation is at most $n\epsilon$ with probability $\exp(-n)$, so we get

$$\left\|\widehat{M}R - \widetilde{M}R\right\|_F \leqslant n^{5}\epsilon$$

which in turn gives

$$\left\|\widetilde{M}R - M\right\|_F \leqslant n^{40}\epsilon,$$

or that M has a good approximation in the column space of $\widetilde{M}$ as well.

On the other hand, Lemma 6.11 gives that $\widetilde{M}$ has full column rank, and has minimum singular value at least ϵn^{-20}. This means that the bit complexity needed to compute $\widehat{X}$ to the specified accuracy of $\widehat{\epsilon}$ is at most $\widehat{\epsilon}\cdot\epsilon\cdot n^{-O(1)}$, or that $\log(\|\|M\|\|_{\infty} n/\epsilon)$ words of precision suffices for these calculations. This gives the round-off error needed when generating S, and in turn the bounds on the word lengths after decimal points in the inputs given to $\textsc{Mult}_M$ and $\textsc{Mult}_{M^T}$.

We then bound the costs of the other steps. First, note that because the max magnitude of entries in S is 1, the max magnitude of entries in $\widehat{M}$ is at most $O(n^2\|\|M\|\|_{\infty})$. Also, because $\widehat{X}$ is an orthonormal basis, the max magnitude of an entry in it is at most 1, which means the max magnitude of an entry in $\widehat{Y}$ is at most $O(n^2\|\|M\|\|_{\infty})$ as well.

Factoring in the round-off errors we incurred gives an additional error of

$$\widehat{\epsilon}\cdot O\left(n^2\|\|M\|\|_{\infty}\right) \leqslant n^5\epsilon.$$

which in turn implies

$$\left\|\widehat{X}\,\widehat{Y}^T - M(r)\right\|_F \leqslant n^{45}\epsilon.$$

with probability at least $1-n^{-10}$, and in turn that the rank r decomposition of $\widehat{X}\,\widehat{Y}^T$ has error at most $n^{45}\epsilon$.

So it remains to bound the overhead caused by running high accuracy singular value decomposition on $\widehat{X}\,\widehat{Y}^T$. Note that the outer product of this matrix is:

$$\left(\widehat{X}\,\widehat{Y}^T\right)^T\widehat{X}\,\widehat{Y}^T = \widehat{Y}\left(\widehat{X}^T\widehat{X}\right)\widehat{Y},$$

and the middle matrix can be computed in $O(nr^{\omega-1})$ operations involving numbers whose magnitude are at most $O(n^{10}\|\|M\|\|_{\infty})$ and with at most $O(\log(n(1+\|\|M\|\|_{\infty})/\epsilon)) + L_M)$ words after decimal place.

Taking square root of this positive semi-definite matrix, and rounding to error

$$\frac{\epsilon}{\|\|M\|\|_{\infty} n^{10}}$$

then allows us to write this matrix as

$$\left(P\widehat{Y}\right)^T P\widehat{Y}$$

for some positive semi-definite matrix $P = (\widehat{X}^T\widehat{X})^{1/2}$. Note that the magnitude of $P\widehat{Y}$ is less than the magnitude of Y due to square-rooting only decreasing eigenvalues above 1.

The equivalence of SVDs of outer and inner products means we can take the SVD of $\widehat{Y}^T P^T P\widehat{Y}$, which is an r-by-r matrix. The complexity of this step was shown to be $\widetilde{O}(r^\omega)$ times the lengths of the initial words in Sections 6.1. and 6.2. of [DDH07]. [3]

It can then be converted back to rank k bases for $P\widehat{Y}$, and then $\widehat{X}$ by multiplying through via these matrices: each of which incurrs an error of at most $O(n^2 \|\|M\|\|_\infty^2)$. Thus, it suffices to keep the round-off errors in all of these steps at most

$$\frac{\epsilon}{n^{50}\left(1+\|\|M\|\|_\infty\right)^4}$$

which gives a word-length of at most $\log(n(1+\|\|M\|\|_\infty)/\epsilon)$. Incorporating this, along with the upper bound of all numbers of $O(n^2\|\|M\|\|)$ together with the $O(nr^{\omega-1})$ operations then gives the total cost. $\square$

We remark that we cannot directly return just $\widehat{X}$ and $\widehat{Y}$. Due to the recursive invocation of such routines, such an overhead will accumulate to $O(1)^d$ over d layers of the recursion. Such error would in turn necessitate stopping the recursion earlier, and in turn at least an $n^{o(1)}$ overhead in running time.

Also, note that the eigen-gap we need for the s-by-s matrix is around ϵ: we can get away with using things like matrix exponential operators from [OSV12, SV13] instead of the full eigen-decomposition routine from [DDH07].

On the other hand, we do remark that this ability to compute a low rank factorization in sublinear time is critical to our overall running time gains. In our setting with $\log(\|M\|_\infty/\epsilon) = \widetilde{O}(m)$ and $L_M = \widetilde{O}(m)$, the total cost of computing this factorization is about $s^\omega \cdot m^2$ word operations, so the reduction from n to s is necesasry for gains over $n^\omega \approx (sm)^\omega$.

6.4 Error Propagation

Our analysis of the error of the recursive algorithm given in Figure 7 relies on bounds on extreme singular values of all principle minors of T containing block-prefixes of indices. Such blocking is natural due to $T_{\{\overline{C},\overline{C}\}}$ taking prefixes of blocks.

Below we provide tools needed to handle the accumulation of errors. Lemma 3.1 implies that singular value bound allow us to carry errors between a matrix and its

[3]The actual stated running time in [DDH07] is $O(r^{\omega+\eta})$ for any $\eta > 0$. We observe this bound is $\widetilde{O}(r^\omega)$ operations involving words whose lengths equal to initial matrix entries: the only overhead on the matrix multiplication steps in [DDH07] are from the $O(\log r)$ layers of recursion.

inverse, losing a factor of $poly(n, \alpha^{-1})$ at each step where α is the min singular value. In order to use this though, we need to bound the singular values of all intermediate matrices that arise from such recursion.

Note that all the matrices that we work with consist of a prefix of blocks, Schur complemented onto a suffix of blocks. Formally, we let $\{S\}$ denote a prefix of the block indices, and $\{C\}$ denote the suffix that we Schur complement onto, and want to bound the min/max singular values of the matrix

$$\mathrm{SC}(T_{\{S,S\}}, \{C\}).$$

The given conditions on H transferred to T ensures that $T_{\{S,S\}}$ has good singular values.

We show via the below lemma that this condition ensures that the Schur complement also has good singular values. Once again, although we only work with block-alligned indices, we prove the bounds for any subset of indices.

Lemma 6.13. *If M is a $n \times n$ square full rank matrix, and $C/\overline{C}$ is a split of the indices such that both M and $M_{\overline{C}\overline{C}}$ have singular values in the range $[\sigma_{\min}, \sigma_{\max}]$, then the singular values of $\mathrm{SC}(M, C)$ are in the range $[n^{-2}\sigma_{\min}, n^{10}\sigma_{\max}^{2}\sigma_{\min}^{-1}]$.*

Proof. Since M is square and full rank, the Schur Complement $\mathrm{SC}(M, C)$ is also full rank. Its min singular value is at least n^{-2} times the inverse of the magnitude of the max entry in its inverse. This entry is in turn at most the max magnitude of an entry in M^{-1}, which is at most $\sigma_{\min}^{-1}$.

The max magnitude in $\mathrm{SC}(M, C)$ follows from bounding the max magnitude of

$$M_{C\overline{C}} M_{\overline{C}\overline{C}}^{-1} M_{\overline{C}C}.$$

The magnitude of each entry in these three matrices are $\sigma_{\max}$, $\sigma_{\min}^{-1}$, and $\sigma_{\max}$ respectively, and there are a total of n^4 tuples to consider (because every Schur complement that arise in a recursive Schur complement routine is the Schur complement onto a suffix block sets of a prefix of blocks). □

Lemmas 6.13 along with the given initial conditions on T then gives that all block Schur complements, as well as their block minors, are well conditioned.

Corollary 6.14. *Let H be a block-Hankel matrix with m blocks of size s, entry magnitude at most $1/(sm)$ and min block singular values at least α_H, that is*

$$\sigma_{\min}\left(H_{\{1:i,(m-i+1):m\}}\right), \sigma_{\min}\left(H_{\{(m-i+1):m,1:i\}}\right) \geqslant \alpha_H \qquad \forall 1 \leqslant i \leqslant m.$$

Let T be the Toeplitz matrix T formed by reversing the order of the column blocks of H. For every leading prefix of block coordinates $\{S\}$, and every trailing blocks of $\{S\}$, $\{C\}$, the Schur complement $\mathrm{SC}(T_{\{S,S\}}, \{C\})$ has singular values in the range

$$\left[n^{-10}\alpha_H, n^{10}\alpha_H^{-3}\right].$$

Note that this covers all leading block minors of Schur complements as well. Specifically, for $\{C\}$ that's a suffix of entries of $\{S\}$, and $\{\widehat{S}\}$ that's a prefix of $\{C\}$, we have

$$\mathrm{SC}\left(T_{\{S\setminus C\cup\widehat{S},S\setminus C\cup\widehat{S}\}},\left\{\widehat{S}\right\}\right)=\mathrm{SC}\left(T_{\{S,S\}},\{C\}\right)_{\{\widehat{S},\widehat{S}\}},$$

Combining Lemma 3.1 and 6.13 also gives that a small error to M implies small errors to all its Schur complements as well.

Lemma 6.15. *Let M be a full rank square matrix, and C a subset of coordinates (with complement $\overline{C}$) such that both M and $M_{\overline{C}\overline{C}}$ have singular values in the range $[\sigma_{\min},\sigma_{\max}]$. For any approximation $\widetilde{M}$ such that*

$$\left\|M-\widetilde{M}\right\|_F\leqslant\epsilon$$

for some $\epsilon<0.1n^{-10}\sigma_{\min}(\sigma_{\max}/\sigma_{\min})^{-2}$, we have:

$$\left\|\mathrm{SC}\left(M,C\right)-\mathrm{SC}\left(\widetilde{M},C\right)\right\|_F\leqslant n^{30}\left(\sigma_{\max}/\sigma_{\min}\right)^4\epsilon$$

Proof. By Fact 6.7, which gives that inverses of Schur complements are subsets of inverses, we can use Lemma 3.1 to get

$$\left\|\mathrm{SC}\left(M,C\right)^{-1}-\mathrm{SC}\left(\widetilde{M},C\right)^{-1}\right\|_F\leqslant\left\|M^{-1}-\widetilde{M}^{-1}\right\|_F\leqslant10\sigma_{\min}^{-2}\epsilon.$$

On the other hand, inverting the singular value bounds on Schur complements from Lemma 6.13 gives that all singular values of $\mathrm{SC}(M,C)^{-1}$ are in the range

$$\left[n^{-10}\sigma_{\max}^{-2}\sigma_{\min},n^{2}\sigma_{\min}^{-1}\right].$$

The given condition on ϵ then gives

$$10\sigma_{\min}^{-2}\epsilon\leqslant n^{-10}\sigma_{\max}^{-2}\sigma_{\min}.$$

So we can invoke Lemma 3.1 once again with $\mathrm{SC}(M,C)$ as the original matrix, and $\mathrm{SC}(\widetilde{M},C)$ as the perturbed matrix gives an overall error bound of

$$10\cdot\left(n^{10}\sigma_{\max}^{-2}/\sigma_{\min}\right)^{-2}\cdot\left(10\sigma_{\min}^{-2}\epsilon\right)\leqslant n^{30}\cdot\left(\sigma_{\max}/\sigma_{\min}\right)^4\epsilon.$$

□

This kind of compounding necessitates a global bounding of errors. We need to show (inductively) that all matrices entering into all stages of the recursion are close to the corresponding matrices of M. Schur complements on the other hand are just as stable to perturbations.

6.5 Analysis of Overall Recursion

We now analyze the overall algorithm by inductively showing that all the matrices produced are close to their exact versions. Our overall error bound for the recursion is as follows:

Lemma 6.16. *Let T be a matrix s.t. the singular values of any block minor Schur complement are in the range $[\alpha_T, \alpha_T^{-1}]$ for some $\alpha_T < (ms)^{-100}$. For $\epsilon < (ms)^{-1}\alpha_T^{10\log m}$, the output of*

$$X^{INV}, Y^{INV} = \textsc{Solve}\left(m, s, X, Y, \epsilon\right)$$

corresponds to a matrix Z with $s^{-}(Z) = X^{INV}(Y^{INV})^T$ such that

$$\left\|T^{-1} - Z\right\|_F \leqslant \alpha_T^{-10\log m}\epsilon.$$

Proof. The proof is by induction on m. The base case of $m = 1$ follows from the guarantees of (fast) matrix inversion [DDHK07].

The top-left blocks of this matrix given by $\{\overline{C}\}$, as well as their associated block Schur complements, satisfy the same singular value bounds due to them being Schur complements of leading minors of T.

Therefore, by the inductive hypothesis, we get that the matrices produced by the first recursive call on Line 4 gives additive inverse at most which incorporating the ϵ additive error from operator generation from Lemma 6.6 gives

$$\left\|T^{-1}_{\{\overline{C},\overline{C}\}} - Z\left(\overline{C}\right)\right\|_F \leqslant \alpha_T^{-10(\log m-1)}\epsilon,$$

Then since the max entry in $T_{\{\overline{C},C\}}$ $T_{\{C,\overline{C}\}}$, and $T^{-1}_{\overline{C},\overline{C}}$ are at most α_T^{-1}, and the approximate operators for multiplying by them have error at most $\alpha_T^{-10(\log m-1)}\epsilon$, the approximate Schur complement onto $\{C\}$ using this approximate inverse $Z(\overline{C})$ and the approximate multiplication operators in T satisfies

$$\left\|\widetilde{\mathrm{SC}} - \mathrm{SC}\left(T, \{C\}\right)\right\|_F \leqslant 3 \cdot 2\alpha_T^{-1} \cdot \alpha_T^{-10(\log m-1)}\epsilon \leqslant \alpha_T^{-2} \cdot \alpha_T^{-10(\log m-1)}\epsilon.$$

Lemma 6.8 gives that $SC(T, \{C\})$ has s^{+}-displacement rank at most $2s$. So combining with the error transfer from Lemma 6.5 gives that the distance from $s^{+}(\widetilde{\mathrm{SC}})$ to a rank $2s$ matrix is at most

$$(ms)^2 \cdot \alpha_T^{-2} \cdot \alpha_T^{-10(\log m-1)}\epsilon$$

Then the guarantees of the low rank approximation procedure $\textsc{LowRankApprox}$ from Lemma 6.12 gives that the resulting factorization has error bounded by

$$\begin{aligned}
&\left\|X\left(SC\right)Y\left(SC\right)^T - s^{+}\left(\mathrm{SC}\left(T, \{C\}\right)\right)\right\|_F \\
&\qquad \leqslant (ms)^{20} \cdot (ms)^2 \alpha_T^{-2} \cdot \alpha_T^{-10(\log m-1)}\epsilon + (ms)^2 \alpha_T^{-2} \cdot \alpha_T^{-10(\log m-1)}\epsilon \\
&\qquad\qquad \leqslant n^{30}\alpha_T^{-2} \cdot \alpha_T^{-10(\log m-1)}\epsilon.
\end{aligned}$$

Applying the error transfer across displacement given by Lemma 6.5 gives that the approximate Schur complement represented by $X(C)$ and $Y(C)$ produced on Line 5, which we denote as $\widehat{\mathrm{SC}}$ satisfies

$$\left\|\widehat{\mathrm{SC}} - \mathrm{SC}\left(T, \{C\}\right)\right\|_F \leqslant n^{40}\alpha_T^{-2} \cdot \alpha_T^{-10(\log m - 1)}\epsilon \leqslant \alpha_T^{-3} \cdot \alpha_T^{-10(\log m-1)}\epsilon.$$

By assumption, the exact Schur complement $\mathrm{SC}(T, C)$ has all consecutive principle minors and Schur complements with singular values in the range $[\alpha_T, \alpha_T^{-1}]$. So the perturbation statement from Lemma 3.1 gives that as long as $\alpha_T^{-3} \cdot \alpha_T^{-10(\log m-1)}\epsilon < \alpha_T$ (which is met by the initial assumption on ϵ). Thus, we get that the singular values of all block minors of $\widehat{\mathrm{SC}}$ are in the range

$$\left[\left(1 - \frac{1}{m}\right)\alpha_T, \left(1 + \frac{1}{m}\right)\alpha_T^{-1}\right].$$

Since $\frac{1}{2} \leqslant (1 - \frac{1}{m})^{O(\log n)}$ and $(1 + \frac{1}{m})^{O(\log n)} \leqslant 2$, this change in α_T is less than the difference made by going from $\log m$ to $\log m - 1$ in the smaller recursive instance. So the choice of ϵ is satisfactory for the recursive call on $X(C)$ and $Y(C)$ as well.

So applying the inductive hypothesis on the second recursive call on Line 7 gives

$$\left\|Z\left(SC\right) - \widehat{\mathrm{SC}}^{-1}\right\|_F \leqslant \alpha_T^{-10(\log m-1)}\epsilon$$

which compounded with the differences with the exact Schur complement gives

$$\left\|Z\left(SC\right) - \mathrm{SC}\left(T, \{C\}\right)\right\|_F \leqslant \alpha_T^{-4}\alpha_T^{-10(\log m-1)}\epsilon.$$

The result for the inductive case then follows from observing that in all the operators for these approximate inverses that these two block matrices are multiplied with have magnitude at most α_T^{-1} and there are at most three such operators composed. Specifically, we get

$$\left\|T^{-1} - Z\right\| \leqslant 3n^2 \cdot \alpha_T^{-1} \cdot \alpha_T^{-4}\alpha_T^{-10(\log m-1)} \leqslant n^{-100}\alpha_T^{-10\log m},$$

Then applying the low rank approximation guarantees from Lemma 6.12 amplifies this by a factor of n^{20}; and the implicit operator construction from Lemma 6.6 incurs an additive ϵ. Incorporating these errors then gives that the inductive hypothesis holds for m as well. □

This means that all the matrices that arise in intermediate steps are close to their exact versions. It allows us to bound the max-magnitude of all the numbers involved, and in turn the number of digits after decimal places. This gives the total cost in number of word operations, which we also bring back to the original statement for solving a well-conditioned block Hankel matrix.

Proof. (of Theorem 3.8)

The proof takes two steps: we first use bounds on the word complexity of X and Y to provide bounds on the word sizes of all numbers involved/generated, including the implicit operators. We then incorporate these lengths into the running time costs of the routines to obtain bounds on overall running time.

Corollary 6.14 gives that the singular values of a block are in the range $[n^{-10}\alpha_H, n^{10}\alpha_H^{-3}]$. So we can set

$$\alpha_T = \alpha_H^4$$

and obtain errors bounds on the recursion via Lemma 6.16. Specifically, to obtain an overall error of ϵ, we need to reduce it by a factor of

$$\alpha_T^{10\log m} \geqslant \alpha_H^{O(\log m)}.$$

This means that the additive difference in any of the intermediate matrices we get is at most

$$\alpha_T^{-10\log m}\epsilon \leqslant \alpha_T^{-1},$$

which combined with the bounds on the matrices themselves of $O(n^{10}\alpha_T^{-3}) \leqslant \alpha_T^{-4}$ by Lemma 6.13 gives that the max magnitude of a matrix that we pass into Lemma 6.12 to factorize is at most α_T^{-5}. It in turn implies that the magnitude of all factorized matrices (Xs and Ys) are at most α_T^{-6}, and they have at most $O(\log m \log(\alpha_T^{-1}\epsilon^{-1}))$ words after the decimal point. Furthermore, because the initial T has min singular value at least α_H, we can truncate the initial X and Y to α_H^3 without affecting the errors.

The implicit multiplication operators generated using fast convolutions via Lemma 6.6 have magnitude at most $O(\alpha_T^6)$, and at most $O(\log^2 m \log(\alpha_T^{-1}\epsilon^{-1}))$ words after the decimal point. As the implicitly generated operators for $\widetilde{SC}$, and the overall inverse Z only composes together a constant number of such operators, the same bounds also hold for the sizes of the numbers, up to a constant factor in the exponent.

We now use these word size bounds to bound the overall running time. These operators are multiplied against sets of $O(s)$ vectors, each with magnitude at most 1 and $O(\log m \log(\alpha_T/\epsilon))$ words after the decimal point in the implicit low rank approximation procedure described in Lemma 6.12. Applying Lemma 6.6 to each of the operators involved gives that this cost is

$$O\left(m \log^5 m s^\omega \log\left(\alpha_T^{-1}\epsilon^{-1}\right)\right) = \widetilde{O}\left(m s^\omega \log\left(\alpha_T^{-1}\epsilon^{-1}\right)\right),$$

while the overhead cost (from matrix factorizations/ orthogonalizations) is

$$\widetilde{O}\left(m s^\omega \cdot \log m \log\left(\alpha_T^{-1}\epsilon^{-1}\right)\right).$$

Combining these, substituting in $\log(1/\alpha_T) = O(\log(1/\alpha_H))$, and incorporating the additional factor of $O(\log m)$ corresponding to the number of layers of recursion then gives the total construction cost.

The invocation/solve cost then follows from the implicit multiplication guarantee of Lemma 6.6 with the $O(\log^2 m \log(\alpha_H^{-1}\epsilon^{-1}))$ word lengths shown above. $\square$

7 Pad and Solve Step

We now analyze the padding algorithm that takes us from a solver for almost the full space, to the full algorithm for solving systems in A. Our padding step combined with the singular value bound from Theorem 3.7 and the stable block-Hankel matrix solver from Theorem 3.8 gives the solver for symmetric, eigenvalue separated matrices, as stated in Lemma 3.9.

Lemma 3.9. *Let A be an $n \times n$ symmetric positive definite matrix with entries at most $1/n$, and $0 < \alpha_A < n^{-10}$ a parameter such that:*

1. *all eigenvalues of A are at least α_A, and at most α_A^{-1},*
2. *all pairs of eigenvalues of A are separated by at least α_A,*
3. *all entries in A have magnitude at most α_A, and at most $O(\log(1/\alpha_A))$ words after the decimal point.*

For any parameter m such that $n^{0.01} \leqslant m \leqslant 0.01n^{0.2}$, the routine BlockKrylov *as shown in Figure 2 pre-processes A in time:*

1. *$O(n)$ matrix-vector multiplications of A against vectors with at most $O(m\log(1/\alpha_A))$ words both before and after the decimal point,*
2. *plus operations that cost a total of:*
$$\widetilde{O}\left(n^2 \cdot m^3 \cdot \log^2(1/\alpha_A) + n^\omega m^{2-\omega} \log(1/\alpha_A)\right).$$

and obtains a routine Solve$_A$ *that when given a vector b with L_b words after the decimal point, returns $Z_A b$ in time*
$$\widetilde{O}\left(n^2 m \cdot \left(\log(1/\alpha_A) + \|b\|_\infty + L_b\right)\right),$$
for some Z_A with at most $O(\log(1/\alpha_A))$ words after the decimal point such that
$$\left\|Z_A - A^{-1}\right\|_F \leqslant \alpha_A.$$

The algorithm is by taking the K generated from Theorem 3.7, and padding a number of random columns to it until it becomes full rank. We first check that we can obtain a bound on the condition number of the overall matrix.

Lemma 7.1. *If $M \in \Re^{n\times d}$ is a n-by-d matrix with $d < n$, max entry magnitude at most $1/n$, and minimum singular value at least α_M, then the n-by-$(d+1)$ matrix formed by appending a dense length n Gaussian vector scaled down by $\frac{1}{n^2}$:*
$$[M, g] \qquad \text{with } g \sim \frac{1}{n^2} \cdot \mathcal{N}(0,1)^n$$
has maximum magnitude at most $1/n$, and minimum singular value at least $n^{-10}\alpha_M$ with probability at least $1 - n^{-2}$.

Proof. By Claim 3.3, we may assume that all entries of g have magnitude at most $1/n$.

We now bound the minimum singular value.

Because $d < n$, there is some unit vector v that's normal to all d columns of A. Consider $d^T g$: by anti-concentration of Gaussians, with probability at least $1 - n^{-3}$ we have

$$\left|d^T g\right| \geqslant n^{-5}.$$

We claim in this case, the matrix $[M, g]$ has minimum singular value at least $n^{-10}\alpha_M$. Consider any test vector $x \in \Re^{d+1}$: if the last entry x_{d+1} has absolute value less than $n^{-2}\alpha_M/10$, then invoking the min-singular value bound on $Ax_{1:d}$ gives

$$\left\|Mx_{1:d}\right\|_2 \geqslant \alpha_M \left\|x_{1:d}\right\|_2 \geqslant \alpha_M \left(1 - |x_{d+1}|\right) \geqslant \frac{\alpha_M}{2}.$$

Then by triangle inequality we get:

$$\left\|[M, g]\, x\right\|_2 \geqslant \left\|Mx_{1:d}\right\|_2 - \left\|gx_{d+1}\right\|_2 \geqslant \frac{\alpha_M}{2} - n^2 \cdot n^{-2}\alpha_M/10 \geqslant \frac{\alpha_M}{10}.$$

So it remains to consider the case where $|x_{d+1}| > n^{-2}\alpha_M/10$. Here because the last column has dot at least n^{-2} against the normal of the previous d columns of M, we get

$$\left\|[M, g]\, x\right\|_2 \geqslant \min_{x_{1:d}} \left\|gx_{d+1} - Mx_{1:d}\right\|_2 = |x_{d+1}| \cdot \left\|g - Mx_{1:d}\right\|_2 = |x_{d+1}| \left\langle g, v\right\rangle \geqslant n^{-7}\alpha_M/10.$$

$\square$

This allows us to bound the condition number of the overall operator.

We also need to account for the costs of computing the blocks of $(AK)^T AK$, as well as performing matrix multiplications in Q and Q^T.

Lemma 7.2. *Given implicit matrix-vector product access to an n-by-n matrix A whose entries have max magnitude at most α_A^{-1}, and at most $O(\log(1/\alpha_A))$ words after the decimal place (for some $0 < \alpha_A \leqslant n^{-10}$), as well as an n-by-s matrix B with at most $nnz(B)$ non-zero entries, each with magnitude at most $\|\|B\|\|_\infty$ and at most L_B digits after the decimal place, as well as m such that $ms \leqslant O(n)$, we can compute the matrix*

$$K = \left[\ B \mid A^1 B \mid A^2 B \mid \ldots \mid A^{m-1} B\ \right]$$

at with the cost of multiplying A against n vectors, each with at most $O(m \log(1/\alpha_A) + \log \|\|B\|\|_\infty + L_B)$ digits, as well as all blocks in its Gram matrix

$$B^T A^i B \qquad 0 \leqslant i \leqslant 2m$$

with additional cost $O(n \cdot nnz(B) \cdot (O(m \log(1/\alpha_A) + \log \|\|B\|\|_\infty + L_B) \log n)$.

Proof. By repeated powering, we can compute $A^i B$ via

$$A^i B = A \cdot A^{i-1} B.$$

Each such powering increases the number of digits before and after the decimal place by at most $O(\log(1/\alpha_A))$, so combining this over the m steps gives the bound.

Note that the same also works for computing all the way up to $A^{2m}B$. Then we need to compute B^T times each of these matrices. This cost is s multiplications of length $O(m\log(1/\alpha_A) + \log(\|\!|B|\!\|) + L_B)$. Each such multiplication can be done using fast multiplication with another overhead of $O(\log n)$, which gives a cost of

$$O\left(s \cdot nnz\left(B\right) \cdot \left(\log\left(\|\!|B|\!\|_{\infty} / \alpha_A\right) + L_B\right) \log n\right)$$

per value of i, which times the $O(m)$ such values and incorporating $sm \leqslant n$ gives the total. □

This gives the initialization cost. Calling the block Hankel matrix solver, and incorporating the guarantees into an overall operator for the padded matrix then gives the overall running time.

Proof. (Of Lemma 3.9)

By Theorem 3.7, the block Krylov space K has max singular value at most n^2, and min singular value at least $\alpha_K = \alpha_A^{5m}$. Consider the full padded matrix

$$Q = \left[\ K \mid G\ \right] = \left[\ G^S \mid A^1 G^S \mid A^2 G^S \mid \ldots \mid A^{m-1} G^S \mid AG\ \right]$$

where $G \in \Re^{n \times (n-ms)}$ is a dense Gaussian matrix, and G^S is a sparse Gaussian matrix with entries set i.i.d to Gaussians with probability $O(\frac{m^2 \log(1/\alpha_A)}{n})$.

Since G has $O(m)$ columns, applying Lemma 7.1 inductively to the extra columns gives that Q has max singular value at most n^2, and min singular value at least

$$\alpha_Q \geqslant \alpha_K \cdot n^{-10 \cdot 10m} \geqslant \alpha_A^{10m}.$$

This in turn means that errors in B by at most α_A^{-20m} will produce error at most $\alpha_A/2$ in the inverse. So we can round all digits in B to such error, obtaining $L_B \leqslant O(m\log(1/\alpha_A))$ as well. Combining this with $\|\!|B|\!\|_{\infty} \leqslant 1$ gives that the cost of the initialization steps given by Lemma 7.2 is n matrix-vector multiplications of A against length n vectors with $O(m\log(1/\alpha_A))$ entries, plus an additional cost of

$$O\left(n \cdot nnz\left(B\right) \cdot m \log\left(1/\alpha_A\right) \log n\right) \leqslant \widetilde{O}\left(n^2 m^3 \log^2\left(1/\alpha_A\right)\right)$$

where the inequality follows from B having $s \leqslant n/m$ columns, each of which have at most $\widetilde{O}(m^3 \log(1/\alpha_A))$ non-zeros with high probability.

Theorem 3.7 gives that with probability at least $1 - n^{-2}$, the min eigenvalue of $(AK)^T AK$ is at least

$$\left(\alpha_A \cdot \alpha_K\right)^2 \geqslant \alpha_A^{20m}.$$

So the block-Hankel solver from Theorem 3.8 with error

$$\epsilon \leftarrow \alpha_A^{1000m}$$

requires construction time:

$$\widetilde{O}\left(ms^{\omega}\cdot\log\left(\alpha_H^{-1}\epsilon^{-1}\right)\right) \leqslant \widetilde{O}\left(m\left(\frac{n}{m}\right)^{\omega}\cdot m\log\left(1/\alpha_A\right)\right) \leqslant \widetilde{O}\left(n^{\omega}m^{2-\omega}\log\left(1/\alpha_A\right)\right).$$

It gives access to a solver operator gives an operator Z_H such that

$$\left\|Z_H - \left((AK)^T AK\right)^{-1}\right\|_2 \leqslant \alpha_A^{1000m}.$$

Now consider the matrix

$$(AQ)^T AQ = \left[\begin{array}{c|c} AK & AG \end{array}\right]^T \left[\begin{array}{c|c} AK & AG \end{array}\right] = \left[\begin{array}{c|c} (AK)^T AK & (AK)^T AG \\ \hline (AG)^T AK & (AG)^T AG \end{array}\right]$$

The bounds that we have on Q gives that the max and min singular values of this matrix is at most α_A^{-100m} and α_A^{100m} respectively. So we can apply Lemma 3.1 repeatedly to replace the top-left block by Z_H:

1. First, by the eigenvalue bounds on $(AK)^T AK$, we have

$$\left\|Z_H^{-1} - (AK)^T AK\right\|_F \leqslant \alpha_A^{800m},$$

which when block-substituted into the overall formula implies

$$\left\|\left[\begin{array}{c|c} Z_H^{-1} & (AK)^T AG \\ \hline (AG)^T AK & (AG)^T (AG) \end{array}\right] - (AQ)^T AQ\right\|_F \leqslant \alpha_A^{800m}.$$

2. Inverting this again using the eigenvalue bound

$$\left\|\left[\begin{array}{c|c} Z_H^{-1} & (AK)^T AG \\ \hline (AG)^T AK & (AG)^T (AG) \end{array}\right]^{-1} - \left((AQ)^T AQ\right)^{-1}\right\|_F \leqslant \alpha_A^{800m}.$$

This new block matrix can also be further factorized as:

$$\left[\begin{array}{c|c} Z_H^{-1} & (AK)^T AG \\ \hline (AG)^T AK & (AG)^T (AG) \end{array}\right] = \left[\begin{array}{c|c} I & \\ \hline (AG)^T AKZ_H & I \end{array}\right]$$
$$\left[\begin{array}{c|c} Z_H^{-1} & 0 \\ \hline 0 & (AG)^T (AG) - (AG)^T AKZ_H (AK)^T AG \end{array}\right]\left[\begin{array}{c|c} I & Z_H (AK)^T AG \\ \hline 0 & I \end{array}\right]$$

which upon inverting becomes

$$\left[\begin{array}{c|c} Z^{-1} & (AK)^T AG \\ \hline (AG)^T AK & (AG)^T (AG) \end{array}\right] = \left[\begin{array}{c|c} I & -Z_H (AK)^T AG \\ \hline 0 & I \end{array}\right]$$
$$\left[\begin{array}{c|c} Z_H & 0 \\ \hline 0 & \left((AG)^T AG - (AG)^T AKZ_H (AK)^T AG\right)^{-1} \end{array}\right] \left[\begin{array}{c|c} I & \\ \hline -(AG)^T AKZ_H & I \end{array}\right]$$

Recall from the start of Section 2 that our definition of linear algorithms are that they exactly evaluate their corresponding operators. As we have access to Z_H, as well as a multiplications by A, K, and G (and their transposes), we are able to evaluate the first and third term exactly. So the only place where additional errors arise are in the the inverse of the bottom-right block of the middle term. Any error in approximating it will get multiplied by the magnitude of the previous and subsequent matrices. The magnitude of the upper/lower triangluar matrices at the start/end are bounded by

$$1 + \left\|Z_H K^T A^2 G\right\|_2 \leqslant 1 + \|Z_H\|_2 \|K\|_2 \|A\|_2^2 \|G\|_2 \leqslant \alpha_A^{-100m},$$

So it suffices to invert the Schur complement term in the middle, specifically $(AG)^T(AG) - (AG)^T AKZ_H(AK)^T AG$, to an additive accuracy α_A^{500m}.

We thus obtain an operator Z_Q such that

$$\left\|Z_Q - \left((AQ)^T AQ\right)^{-1}\right\|_F \leqslant \alpha_A^{300m}.$$

Finally, note that both Q and $(AQ)^T$ are matrices with magnitude at most α_A^{-100m}, and

$$Q\left((AQ)^T AQ\right)^{-1} (AQ)^T = QQ^{-1}A (AQ)^{-T} (AQ)^T = A^{-1}.$$

Substituting this in then gives:

$$\left\|A^{-1} - QZ_Q (AQ)^T\right\|_2 = \left\|Q\left(\left((AQ)^T AQ\right)^{-1} - Z_Q\right)(AQ)^T\right\|_2$$
$$\leqslant \|Q\|_2 \left\|\left((AQ)^T AQ\right)^{-1} - Z_Q\right\|_F \|AQ\|_2 \leqslant \alpha_A^{100m} \leqslant \alpha_A/n^2,$$

which means the final post-processing step gives the desired error guarantees.

For the bit-complexity of the operator Z_A, Theorem 3.8 gives that the numbers of words after decimal place in Z_H is at most

$$O\left(\log^2 n \log\left(\alpha_H^{-1}\epsilon^{-1}\right)\right) = O\left(m \log^2 n \log\left(1/\alpha_A\right)\right).$$

The matrices A, K, Q have at most $O(m \log(1/\alpha_A))$ words after the decimal place. So because the multiplications only involve a constant number of such operators, the maximum number of digits after the decimal place we'll have is also $O(m \log^2 n \log(1/\alpha_A))$, and the max magnitude of an entry we'll encounter is $\alpha_A^{-O(m)}$.

Thus, the solve costs from Theorem 3.8 needs to be invoked on vectors whose entries have magnitude at most $\alpha_A^{-O(m)} \left\| b \right\|_{\infty}$, and at most $O(L_B + m\log^2 n \log(1/\alpha_A))$ after the decimal point, giving:

$$\widetilde{O}\left(m \cdot s^2 \cdot \left(\log\left(\frac{\left(1+\alpha_A^{-O(m)}\left\| b \right\|_{\infty}\right) n}{\alpha_H \epsilon}\right) + m\log^2 n \log\left(1/\alpha_A\right) + L_B\right)\right)$$
$$\leqslant \widetilde{O}\left(ms^2 \cdot \left(m\log^2 n\log\left(1/\alpha_A\right) + \log\left\| B \right\|_{\infty} + L_B\right)\right)$$
$$\leqslant \widetilde{O}\left(n^2\left(\log\left(1/\alpha_A\right) + \log\left\| B \right\|_{\infty} + L_B\right)\right).$$

On the other hand, this vector with $O(m\log^2 n \log(1/\alpha_A) + \log \left\| B \right\|_{\infty} + L_B)$ words per entry needs to be multiplied against Q and K, which are dense matrices with n^2 entries of word-size at most $O(m\log(1/\alpha_A))$. So the total cost is:

$$\widetilde{O}\left(n^2 m\left(\log\left(1/\alpha_A\right) + \log\left\| B \right\|_{\infty} + L_B\right)\right),$$

which is more than the above term from the Hankel matrix solver due to $ms \leqslant n$ and $m > n^{0.01}$. $\square$

8 Discussion

We have presented a faster solver for linear systems with moderately sparse coefficient matrices under bounded word-length arithmetic with logarithmic dependence on the condition number. This is the first separation between the complexity of matrix multiplication and solving linear systems in the bounded precision setting. While both our algorithm and analysis are likely improvable, we believe they demonstrate that there are still many sparse numerical problems and algorithms that remain to be better understood theoretically. We list a few avenues for future work.

Random Matrices. The asymptotic gap between our running time of about $n^{2.33}$ and the $O(n^{2.28})$ running time of computing inverses of sparse matrices over finite fields [EGG+06] is mainly due to the overhead our minimum singular value bound from Theorem 3.7, specifically the requirement of $\Omega(m^3)$ non-zeros per column on average. We conjecture that a similar bound holds for $\widetilde{O}(m)$ non-zeros per column, and also in the full Krylov space case.

1. Can we lower bound the min singular value of a block Krylov space matrix generated from a random matrix with $\widetilde{O}(m)$ non-zeros per column?

2. Can we lower bound the min singular value of a block Krylov space matrix where $m \cdot s = n$ for general values of s (block size) and m (number of steps)?

The second improvement of removing the additional $\Omega(m)$ columns would not give asymptotic speedups. It would however remove the need for the extra steps (padding with random Gaussian columns) in Section 7. Such a bound for the square case would likely require developing new tools for analyzing matrix anti-concentration.

3. Are there general purpose bounds on the min singular value of a sum of random matrices, akin to matrix concentration bounds (which focus on the max singular value) [RV10, Tro15].

The connections with random matrix theory can also be leveraged in the reverse direction:

4. Can linear systems over random matrices with i.i.d. entries be solved faster?

An interesting case here is sparse matrices with non-zeros set to ± 1 independently. Such matrices have condition number $\Theta(n^2)$ with constant probability [RV10], which means that the conjugate gradient algorithm has bit complexity $O(n \cdot nnz)$ on such systems. Therefore, we believe these matrices present a natural starting point for investigating the possibility of faster algorithms for denser matrices with $nnz > \Omega(n^{\omega-1})$.

Numerical Algorithms. The bounded precision solver for block Hankel matrices in Section 6 is built upon the earliest tools for speeding up solvers for such structured matrices [KKM79, BA80], as well as the first sparsified block Cholesky algorithm for solving graph Laplacians [KLP+16]. We believe the more recent developments in solvers for Hankel/Toeplitz matrices [XXCB14] as well as graph Laplacians [KS16] can be incorporated to give better and more practical routines for solving block-Hankel/Toeplitz matrices.

5. Is there a superfast solver under bounded precision for block Hankel/Toeplitz matrices that does not use recursion?

It would also be interesting to investigate whether recent developments in randomized numerical linear algebra can work for Hankel/Toeplitz matrices. Some possible questoins there are:

6. Can we turn $m \times s^{\omega}$ into $O(ms^2 + s^{\omega})$ using more recent developments sparse projections (e.g. CountSketch / sparse JL / sparse Gaussian instead of a dense Gaussian).
7. Is there an algorithm that takes a rank r factorization of $I - XY \in \Re^{n \times n}$, and computes in time $\widetilde{O}(n\text{poly}(r))$ the a rank r factorization/approximation of $I - YX$?

Another intriguing question is the extensions of this approach to the high condition number, or exact integer solution, setting. Here the current best running time bounds are via p-adic representations of fractions [Dix82], which are significantly less understood compared to decimal point based representations. In the dense case, an algorithm via shifted p-adic numbers by Storjohann [Sto05] achieves an $O(n^{\omega})$ bit complexity. Therefore, it is natural to hope for a similar $\widetilde{O}(n \cdot nnz)$ bit complexity algorithm for producing exact integer solutions. A natural starting point could be the role of low-rank sketching in solvers that take advantage of displacement rank, i.e., extending the p-adic algorithms to handle low rank matrices:

8. Is there an $O(n \cdot r^{\omega-1})$ time algorithm for exactly solving linear regression problems involving an n-by-n integer matrix with rank r?

Finally, we note that the paper by Eberly et al. [EGG$^+$06] that proposed block-Krylov based methods for matrix inversion also included experimental results that demonstrated good performances as an exact solver over finite fields. It might be possible to practically evaluate block-Krylov type methods for solving general systems of linear equations. Here it is worth remarking that even if one uses naive $\Theta(n^3)$ time matrix multiplication, both the Eberly et al. algorithm [EGG$^+$07] (when combined with p-adic representations), as well as our algorithm, still take sub-cubic time.

Acknowldgements

Richard Peng was supported in part by NSF CAREER award 1846218, and Santosh Vempala by NSF awards AF-1909756 and AF-2007443. We thank Mark Giesbrecht for bringing to our attention the literature on block-Krylov space algorithms; Yin Tat Lee for discussions on random linear systems; Yi Li, Anup B. Rao, and Ameya Velingker for discussions about high dimensional concentration and anti-concentration bounds; Mehrdad Ghadiri, He Jia, Kyle Luh, Silvia Casacuberta Puig, and anonymous reviewers for comments on earlier versions of this paper.

References

[BA80] Robert R Bitmead and Brian DO Anderson. Asymptotically fast solution of Toeplitz and related systems of linear equations. *Linear Algebra and its Applications*, 34:103–116, 1980.

[BBV04] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[BHK20] Avrim Blum, John Hopcroft, and Ravindran Kannan. *Foundations of Data Science*. Cambridge University Press, 2020.

[BHV08] Erik G. Boman, Bruce Hendrickson, and Stephen A. Vavasis. Solving elliptic finite element systems in near-linear time with support preconditioners. *SIAM J. Numer. Anal.*, 46(6):3264–3284, 2008.

[BJMS17] Alin Bostan, C-P Jeannerod, Christophe Mouilleron, and É Schost. On matrices with displacement structure: Generalized operators and faster algorithms. *SIAM Journal on Matrix Analysis and Applications*, 38(3):733–775, 2017.

[BL94] Bernhard Beckermann and George Labahn. A uniform approach for the fast computation of matrix-type padé approximants. *SIAM Journal on Matrix Analysis and Applications*, 15(3):804–823, 1994.

[Blu04] Lenore Blum. Computing over the reals: Where turing meets newton. *Notices of the AMS*, 51(9):1024–1034, 2004.

[BSS$^+$89] Lenore Blum, Mike Shub, Steve Smale, et al. On a theory of computation and complexity over the real numbers: *np*-completeness, recursive functions and universal machines. *Bulletin (New Series) of the American Mathematical Society*, 21(1):1–46, 1989.

[BT87] Jean Bourgain and Lior Tzafriri. Invertibility of 'large' submatrices with applications to the geometry of Banach spaces and harmonic analysis. *Israel Journal of Mathematics*, 57(2):137–224, 1987.

[Cip00] Barry A Cipra. The best of the 20th century: Editors name top 10 algorithms. *SIAM news*, 33(4):1–2, 2000. Available at: https://archive.siam.org/pdf/news/637.pdf.

[CKK$^+$18] Michael B. Cohen, Jonathan A. Kelner, Rasmus Kyng, John Peebles, Richard Peng, Anup B. Rao, and Aaron Sidford. Solving directed laplacian systems in nearly-linear time through sparse LU factorizations. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 898–909. IEEE Computer Society, 2018. Available at: https://arxiv.org/abs/1811.10722.

[CKP$^+$17] Michael B. Cohen, Jonathan A. Kelner, John Peebles, Richard Peng, Anup B. Rao, Aaron Sidford, and Adrian Vladu. Almost-linear-time algorithms for markov chains and new spectral primitives for directed graphs. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 410–419. ACM, 2017. Available at: https://arxiv.org/abs/1611.00755.

[CLRS09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, 3rd Edition.* MIT Press, 2009.

[CRW93] Ronald Coifman, Vladimir Rokhlin, and Stephen Wandzura. The fast multipole method for the wave equation: A pedestrian prescription. *IEEE Antennas and Propagation magazine*, 35(3):7–12, 1993.

[CW87] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 1–6. ACM, 1987.

[DDH07] James Demmel, Ioana Dumitriu, and Olga Holtz. Fast linear algebra is stable. *Numerische Mathematik*, 108(1):59–91, 2007.

[DDHK07] James Demmel, Ioana Dumitriu, Olga Holtz, and Robert Kleinberg. Fast matrix multiplication is stable. *Numerische Mathematik*, 106(2):199–224, 2007.

[Dix82] John D Dixon. Exact solution of linear equations using P-adic expansions. *Numerische Mathematik*, 40(1):137–141, 1982.

[DMM08] 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. Available at:.

[DRSL16] Timothy A Davis, Sivasankaran Rajamanickam, and Wissam M Sid-Lakhdar. A survey of direct methods for sparse linear systems. *Acta Numerica*, 25:383–566, 2016.

[EGG$^+$06] Wayne Eberly, Mark Giesbrecht, Pascal Giorgi, Arne Storjohann, and Gilles Villard. Solving sparse rational linear systems. In *Symbolic and Algebraic Computation, International Symposium, ISSAC 2006, Genoa, Italy, July 9-12, 2006, Proceedings*, pages 63–70, 2006.

[EGG$^+$07] Wayne Eberly, Mark Giesbrecht, Pascal Giorgi, Arne Storjohann, and Gilles Villard. Faster inversion and other black box matrix computations using efficient block projections. In *Symbolic and Algebraic Computation, International Symposium, ISSAC 2007, Waterloo, Ontario, Canada, July 28 - August 1, 2007, Proceedings*, pages 143–150, 2007.

[EH10] Herbert Edelsbrunner and John Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.

[GK72] I Gohberg and N Ya Krupnik. A formula for the inversion of finite toeplitz matrices. *Mat. Issled*, 7(12):272–283, 1972.

[GO89] Gene H Golub and Dianne P O'Leary. Some history of the conjugate gradient and lanczos algorithms: 1948–1976. *SIAM review*, 31(1):50–102, 1989.

[Gra06] Robert M Gray. *Toeplitz and circulant matrices: A review*. now publishers inc, 2006.

[Gre96] Keith D. Gremban. *Combinatorial Preconditioners for Sparse, Symmetric, Diagonally Dominant Linear Systems*. PhD thesis, Carnegie Mellon University, Pittsburgh, October 1996. CMU CS Tech Report CMU-CS-96-123.

[GTVDV96] KA Gallivan, S Thirumalai, Paul Van Dooren, and V Vermaut. High performance algorithms for toeplitz and block toeplitz matrices. *Linear algebra and its applications*, 241:343–388, 1996.

[HS52] Magnus Rudolph Hestenes and Eduard Stiefel. *Methods of conjugate gradients for solving linear systems*, volume 49-1. NBS Washington, DC, 1952.

[HVDH19] David Harvey and Joris Van Der Hoeven. Polynomial multiplication over finite fields in time $o(nlogn)$. Available at `https://hal.archives-ouvertes.fr/hal-02070816/document`, 2019.

[KKM79] Thomas Kailath, Sun-Yuan Kung, and Martin Morf. Displacement ranks of matrices and linear equations. *Journal of Mathematical Analysis and Applications*, 68(2):395–407, 1979.

[KLP$^+$16] Rasmus Kyng, Yin Tat Lee, Richard Peng, Sushant Sachdeva, and Daniel A Spielman. Sparsified cholesky and multigrid solvers for connection laplacians. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 842–850. ACM, 2016. Available at http://arxiv.org/abs/1512.01892.

[KMP12] Ioannis Koutis, Gary L. Miller, and Richard Peng. A fast solver for a class of linear systems. *Communications of the ACM*, 55(10):99–107, October 2012. Available at https://cacm.acm.org/magazines/2012/10/155538-a-fast-solver-for-a-class-of-linear-systems/fulltext.

[KS16] Rasmus Kyng and Sushant Sachdeva. Approximate gaussian elimination for laplacians - fast, sparse, and simple. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 573–582, 2016. Available at: https://arxiv.org/abs/1605.02353.

[KV17] Ravindran Kannan and Santosh Vempala. Randomized algorithms in numerical linear algebra. *Acta Numerica*, 26:95, 2017.

[KWZ20] Rasmus Kyng, Di Wang, and Peng Zhang. Packing lps are hard to solve accurately, assuming linear equations are hard. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 279–296. SIAM, 2020. Available at: https://dl.acm.org/doi/pdf/10.5555/3381089.3381106.

[Kyn17] Rasmus Kyng. *Approximate Gaussian Elimination.* PhD thesis, Yale University, 2017. Available at: `http://rasmuskyng.com/rjkyng-dissertation.pdf`.

[KZ17] Rasmus Kyng and Peng Zhang. Hardness results for structured linear systems. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 684–695, 2017. Available at: https://arxiv.org/abs/1705.02944.

[Lan50] 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*, 1950.

[LG14] Francois Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation*, pages 296–303. ACM, 2014. Available at http://arxiv.org/abs/1401.7714.

[LL19] Patrick Lopatto and Kyle Luh. Tail bounds for gaps between eigenvalues of sparse random matrices. *arXiv preprint arXiv:1901.05948*, 2019.

[LLY11] Lin Lin, Jianfeng Lu, and Lexing Ying. Fast construction of hierarchical matrix representation from matrix–vector multiplication. *Journal of Computational Physics*, 230(10):4071–4087, 2011.

[LS92] George Labahn and Tamir Shalom. Inversion of toeplitz matrices with only two standard equations. *Linear algebra and its applications*, 175:143–158, 1992.

[LV18] Kyle Luh and Van Vu. Sparse random matrices have simple spectrum. *arXiv preprint arXiv:1802.03662*, 2018.

[MMS18] Cameron Musco, Christopher Musco, and Aaron Sidford. Stability of the lanczos method for matrix function approximation. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1605–1624, 2018. Available at: https://arxiv.org/abs/1708.07788.

[NTV17] Hoi Nguyen, Terence Tao, and Van Vu. Random matrices: tail bounds for gaps between eigenvalues. *Probability Theory and Related Fields*, 167(3-4):777–816, 2017.

[OSV12] Lorenzo Orecchia, Sushant Sachdeva, and Nisheeth K. Vishnoi. Approximating the exponential, the Lanczos method and an $\widetilde{O}(m)$-time spectral algorithm for balanced separator. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 1141–1160. ACM, 2012.

[Pan84] Victor Y. Pan. *How to Multiply Matrices Faster*, volume 179 of *Lecture Notes in Computer Science*. Springer, 1984.

[PS85] Franco P Preparata and Michael Ian Shamos. *Computational geometry. an introduction*. Springer-Verlag New York, 1985.

[RV07] Mark Rudelson and Roman Vershynin. Sampling from large matrices: An approach through geometric functional analysis. *J. ACM*, 54(4):21, 2007.

[RV10] Mark Rudelson and Roman Vershynin. Non-asymptotic theory of random matrices: extreme singular values. In *Proceedings of the International Congress of Mathematicians 2010 (ICM 2010) (In 4 Volumes) Vol. I: Plenary Lectures and Ceremonies Vols. II–IV: Invited Lectures*, pages 1576–1602. World Scientific, 2010.

[Saa03] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd edition, 2003. Available at http://www-users.cs.umn.edu/~saad/toc.pdf.

[Sar06] Tamás Sarlós. Improved approximation algorithms for large matrices via random projections. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 143–152. IEEE Computer Society, 2006.

[SS12] Daniel A Spielman and Nikhil Srivastava. An elementary proof of the restricted invertibility theorem. *Israel Journal of Mathematics*, 190(1):83–91, 2012. Available at https://arxiv.org/pdf/0911.1114.pdf.

[SST06] Arvind Sankar, Daniel A Spielman, and Shang-Hua Teng. Smoothed analysis of the condition numbers and growth factors of matrices. *SIAM Journal on Matrix Analysis and Applications*, 28(2):446–476, 2006.

[ST11] D. Spielman and S. Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011. Available at http://arxiv.org/abs/0808.4134.

[ST14] D. Spielman and S. Teng. Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM Journal on Matrix Analysis and Applications*, 35(3):835–885, 2014. Available at http://arxiv.org/abs/cs/0607105.

[Sto05] Arne Storjohann. The shifted number system for fast linear algebra on integer matrices. *Journal of Complexity*, 21(4):609–650, 2005. Available at: https://cs.uwaterloo.ca/˜astorjoh/shifted.pdf.

[Str69] Volker Strassen. Gaussian elimination is not optimal. *Numerische mathematik*, 13(4):354–356, 1969.

[SV13] Sushant Sachdeva and Nisheeth K Vishnoi. Faster algorithms via approximation theory. *Theoretical Computer Science*, 9(2):125–210, 2013.

[Tro15] Joel A. Tropp. An introduction to matrix concentration inequalities. *Found. Trends Mach. Learn.*, 8(1-2):1–230, 2015.

[Tur48] Alan M Turing. Rounding-off errors in matrix processes. *The Quarterly Journal of Mechanics and Applied Mathematics*, 1(1):287–308, 1948.

[TV10] Terence Tao and Van H. Vu. Smooth analysis of the condition number and the least singular value. *Math. Comput.*, 79(272):2333–2352, 2010. Available at: https://arxiv.org/abs/0805.3167.

[Vai89] P. M. Vaidya. Speeding-up linear programming using fast matrix multiplication. In *30th Annual Symposium on Foundations of Computer Science*, pages 332–337, Oct 1989.

[Wil12] Virginia Vassilevska Williams. Multiplying matrices faster than coppersmith-winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 887–898. ACM, 2012.

[Woo14] David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014. Available at: `https://arxiv.org/abs/1411.4357`.

[XB90] Guo-liang Xu and Adhemar Bultheel. Matrix padé approximation: definitions and properties. *Linear algebra and its applications*, 137:67–136, 1990.

[XCGL10] Jianlin Xia, Shivkumar Chandrasekaran, Ming Gu, and Xiaoye S Li. Fast algorithms for hierarchically semiseparable matrices. *Numerical Linear Algebra with Applications*, 17(6):953–976, 2010.

[XXCB14] Yuanzhe Xi, Jianlin Xia, Stephen Cauley, and Venkataramanan Balakrishnan. Superfast and stable structured solvers for toeplitz least squares via randomized sampling. *SIAM Journal on Matrix Analysis and Applications*, 35(1):44–72, 2014.

[XXG12] Jianlin Xia, Yuanzhe Xi, and Ming Gu. A superfast structured solver for toeplitz linear systems via randomized sampling. *SIAM Journal on Matrix Analysis and Applications*, 33(3):837–858, 2012.

[Ye11] Yinyu Ye. *Interior point algorithms: theory and analysis*, volume 44. John Wiley & Sons, 2011.

[Zha18] Peng Zhang. *Hardness and Tractability For Structured Numerical Problems.* PhD thesis, Georgia Institute of Technology, 2018. Available at: `https://drive.google.com/file/d/1KEZNzna-Y7y6rDERKuBFvFuU-hfHUMR3/view`.

A Properties of Vandermonde Matrices

We prove the large entries property of Vandermonde matrices.

Lemma 5.1. *Let σ be a length n vector with all entries in the range $[\alpha, \alpha^{-1}]$ for some $\alpha < 1/n$, and any two entries of σ at least α apart. The $n \times m$ Vandermonde matrix with entries given as*

$$V_{ij} = \sigma_i^{j-1}.$$

has the property that for any unit vector x, at least $n-m+1$ entries in Vx have magnitude at least α^{3m}.

We first prove the square case.

Lemma A.1. *Let $0 < \alpha < 1$ be a bound and $\alpha < \lambda_1 \leqslant \lambda_2 \leqslant \ldots \leqslant \lambda_m \leqslant \alpha^{-1}$ be positive real numbers such that $\lambda_{i+1} - \lambda_i \geqslant \alpha$, then the Vandermonde matrix*

$$V = \begin{bmatrix} 1 & \lambda_1 & \lambda_1^2 & \ldots & \lambda_1^{m-1} \\ 1 & \lambda_2 & \lambda_2^2 & \ldots & \lambda_2^{m-1} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & \lambda_m & \lambda_m^2 & \ldots & \lambda_m^{m-1} \end{bmatrix}$$

is full rank, and has minimum singular value at least $m^{-1}2^{-m}\alpha^m$.

Proof. We will bound the minimum singular value by bounding the maximum magnitude of an entry in $V(\lambda, m)^{-1}$. We will in turn bound this by explicitly writing down the inverse using Lagrange interpolation of polynomials.

For a vector $x = [x_1; x_2; \ldots; x_m]$, consider the polynomial in t with coefficients given by x, specifically

$$p_x(t) = \sum_{i=1}^{m} x_i t^{i-1}.$$

The matrix-vector product Vx can be viewed as evaluating $p_x(\cdot)$ at $\lambda_1, \lambda_2, \ldots \lambda_m$:

$$Vx = [p_x(\lambda_1); p_x(\lambda_2); \ldots; p_x(\lambda_m)].$$

This means that if we are given these polynomial values as a vector b, we can solve for the value of x using the Lagrange interpolation formula

$$p_x(t) = \sum_{i=1}^{m} b_i \cdot \prod_{j \neq i} \frac{(t - \lambda_j)}{(\lambda_i - \lambda_j)}.$$

Thus the inverse is formed by collecting coefficients on each of the monomial t^is. To do this, consider expanding each of the product terms. Since $\lambda_j \leqslant \alpha^{-1}$, each of the numerator's contribution is at most α^{-m}. Also, since $|\lambda_i - \lambda_j| \geqslant \alpha$, the denominator contributes at most α^{-1} as well.

Coupled with the fact that there are at most 2^m terms in expansions, each entry of inverse has magnitude at most $2^m\alpha^{-2m}$. Plugging in any unit vector x then gives

$$\left\|x\right\|_2 \leqslant \left\|V^{-1}\right\|_2 \left\|Vx\right\|_2,$$

which plugging in $\left\|V^{-1}\right\|_2 \leqslant m \cdot 2^m\alpha^{-2m}$ above gives $\left\|Vx\right\|_2 \geqslant m^{-1}2^{-m}\alpha^{2m}$. ☐

For the rectangular case, we simply apply Lemma A.1 to every subset of m rows.

Proof. (of Lemma 5.1) If there is some unit x such that Vx has m entries with magnitude less than α^{3m}, then let this subset be S. We have

$$\left\|V_{S,:}x\right\|_2 \leqslant m^{-1}2^{-m}\alpha^{2m}.$$

On the other hand, because λ_S is a subset of λ, it is also in the range $[\alpha, \alpha^{-1}]$, is separated by at least α. So we get a contradiction with Lemma A.1. ☐

B Properties and Algorithms for Low Displacement Rank Matrices

We first check the preservation of displacement rank (up to a sign/direction flip) under inversion.

Lemma 6.4. *For any invertible matrix M and any shift value s, we have*

$$\textsc{Rank}_{+s}\left(M\right) = \textsc{Rank}_{-s}\left(M^{-1}\right)$$

The proof of this lemma relies on the following generalization of 'left inverse is right inverse'.

Fact B.1. *For any square (but not necessarily invertible) matrices $M^{(1)}$ and $M^{(2)}$, we have*

$$\textsc{Rank}\left(I - M^{(1)}M^{(2)}\right) = \textsc{Rank}\left(I - M^{(2)}M^{(1)}\right).$$

Proof. We will show $\textsc{Rank}(I - M^{(1)}M^{(2)}) \geqslant \textsc{Rank}(I - M^{(2)}M^{(1)})$, or $\textsc{Null}(I - M^{(1)}M^{(2)}) \leqslant \textsc{Null}(I - M^{(2)}M^{(1)})$. The other direction follows from flipping the role of $M^{(1)}$ and $M^{(2)}$ and applying the argument again.

Let dimension of the null space of $I - M^{(1)}M^{(2)}$ be r. Let $x^{(1)}, x^{(2)}, \ldots x^{(r)}$ be a basis for the null space of $I - M^{(1)}M^{(2)}$. The condition

$$\left(I - M^{(1)}M^{(2)}\right) x^{(\hat{r})} = 0 \qquad \forall 1 \leqslant \hat{r} \leqslant r$$

rearranges to

$$x^{(\widehat{r})} = M^{(1)} M^{(2)} x^{(\widehat{r})} \qquad \forall 1 \leqslant \widehat{r} \leqslant r$$

then for each $\widehat{r}$, $M^{(2)} x^{(\widehat{r})}$ is also in the null space of $I - M^{(2)} M^{(1)}$:

$$\left(I - M^{(2)} M^{(1)}\right) M^{(2)} x^{(\widehat{r})} = M^{(2)} x^{(\widehat{r})} - M^{(2)} M^{(1)} M^{(2)} x^{(\widehat{r})} = M^{(2)} x^{(\widehat{r})} - M^{(2)} x^{(\widehat{r})} = 0$$

So it remains to show that $M^{(2)} x^{(\widehat{r})}$s are linearly independent. Note that because $M^{(2)}$ may not be invertible, we cannot just conclude that $x^{(\widehat{r})}$ are themselves linearly independent. Instead, we need to use the fact that $x^{(\widehat{r})}$ is invariant under $M^{(1)} M^{(2)}$ to 'locally invert' $M^{(2)} x^{(\widehat{r})}$ using $M^{(1)}$. Formally, suppose $c_1 \ldots c_r$ not all 0 are coefficients such that

$$\sum_{1 \leqslant \widehat{r} \leqslant r} c_{\widehat{r}} M^{(2)} x^{(\widehat{r})} = 0$$

then

$$0 = M^{(1)} \left(\sum_{1 \leqslant \widehat{r} \leqslant r} c_{\widehat{r}} M^{(2)} x^{(\widehat{r})} \right) = \sum_{1 \leqslant \widehat{r} \leqslant r} c_{\widehat{r}} M^{(1)} M^{(2)} x^{(\widehat{r})} = \sum_{1 \leqslant \widehat{r} \leqslant r} c_{\widehat{r}} x^{(\widehat{r})},$$

a contradiction with the assumption that $x^{(\widehat{r})}$s are linearly independent. ☐

Proof. (of Lemma 6.4) Because M is square and full rank, the rank of a matrix is preserved when multiplying by M and M^{-1}.

$$\alpha_{-}\left(M^{-1}\right) = \textsc{Rank}\left(M^{-1} - \Delta(s)^T M^{-1} \Delta(s)\right) = \textsc{Rank}\left(I - \Delta(s)^T M^{-1} \Delta(s) M\right).$$

Invoking Fact B.1 on the matrices $\Delta(s)^T M^{-1}$ and $\Delta(s) M$ then gives

$$\alpha_{-}\left(M^{-1}\right) = \textsc{Rank}\left(I - \Delta(s) M \Delta(s)^T M^{-1}\right)$$

which when multiplied by M gives

$$= \textsc{Rank}\left(M - \Delta(s) M \Delta(s)^T\right) = \alpha_{+}(M).$$

☐

Next we check that representations in the displaced version can be used to efficiently perform matrix-vector multiplications in the original matrix.

Lemma 6.6. *Given block size s, block count m, (ms)-by-r matrices X and Y, the (ms)-by-(ms) matrix M such that*

$$M - \Delta(s) M \Delta(s)^T = X^T Y$$

has a unique solution.

Furthermore, there is a routine ImplicitMatVec *that for any accuracy $\delta < (ms)^{-10}$ corresponds to a linear operator $\widetilde{Z}_{XY\to M,\delta}$ such that for any ms-by-k matrix B with at most L_B words after decimal place,* ImplicitMatVec(X, Y, B, δ) *takes time (measured in number of word operations)*

$$O\left(m\log^3 m \cdot \max\{r,s\}\max\left\{s^{\omega-2}k, sk^{\omega-2}\right\}\right.$$
$$\left.\cdot\left(L_B + \log\left(\left(1+\|X\|_\infty\right)\left(1+\|Y\|_\infty\right)\left(1+\|B\|_\infty\right)ms/\delta\right)\right)\right)$$

and outputs the (ms)-by-k matrix

$$\widetilde{Z}_{XY\to M,\delta}B$$

where $\widetilde{Z}_{XY\to M,\delta}$ is a matrix with at most $O(\log m \log((1+\|X\|_\infty)(1+\|Y\|_\infty)ms/\delta))$ words after the decimal point such that

$$\left\|\widetilde{Z}_{XY\to M,\delta} - M\right\|_F \leqslant \delta$$

The same guarantees and runtime bounds also hold for the negative displacement case where M is implicitly specified as $M - \Delta(s)^T M\Delta(s) = XY^T$, as well as matrix-vector multiplications with M^T.

Our goal is to use matrix-vector convolutions. To do this, we first represent M as the product of upper and lower (block) triangular matrices that are also Toeplitz: every diagonal consists of the same blocks. This property is in turn useful because multiplying by these special types of matrices can in turn be carried using fast convolutions.

For notational simplicity (specifically to avoid using U for upper-triangular matrices), we only use lower triangular forms of these, which we denote T_L.

Definition B.2. For an $(ms) \times s$ matrix X, the corresponding block lower-triangular Toeplitz matrix $T_L(X)$ is given by placing X on the leftmost column,

$$T_L(X) = \begin{bmatrix} X_{\{1,1\}} & 0 & 0 & \dots & 0 \\ X_{\{2,1\}} & X_{\{1,1\}} & 0 & \dots & 0 \\ X_{\{3,1\}} & X_{\{2,1\}} & X_{\{1,1\}} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ X_{\{m,1\}} & X_{\{m-1,1\}} & X_{\{m-2,1\}} & \dots & X_{\{1,1\}} \end{bmatrix}$$

or formally

$$T_L(X)_{\{i,j\}} = \begin{cases} X_{\{i-j+1,1\}} & \text{if } j \leqslant i, \\ 0 & \text{otherwise.} \end{cases}$$

It was shown in [KKM79] that the product applying $T_L(\cdot)$ to the two factors gives a representation of the original displacement matrix. Note that both the displacement operation, and the computation of XY^T is linear in the matrices. Therefore, we can view X and Y as t copies of $n \times s$ matrices, with possibly padding by 0s to make their second dimension a multiple of s.

Lemma B.3. *For any number t and any sequence of $(ms) \times s$ matrices $X^{(1\ldots t)}$ and $Y^{(1\ldots t)}$, the matrix $M^{(+)}$ satisfying the equation*

$$+s\left(M^{(+)}\right) = \sum_{\hat{t}=1}^{t} X^{(\hat{t})}\left(Y^{(\hat{t})}\right)^T$$

has the unique solution

$$M^{(+)} = \sum_{\hat{t}=1}^{t} T_L\left(X^{(\hat{t})}\right) T_L\left(Y^{(\hat{t})}\right)^T$$

and similarly the unique solution to $s - (M^{(-)}) = XY^T$ is

$$M^{(-)} = \sum_{\hat{t}=1}^{t} T_L\left(X^{(\hat{t})}\right)^T T_L\left(Y^{(\hat{t})}\right)$$

Proof. By symmetry (in reversing both the order of rows and columns), we focus on the case with $M^{(+)}$. We first show that the solution is unique.

Since $\Delta(s)$ shifts all entries down by s, we have

$$\left(M - \Delta\left(s\right) M \Delta\left(s\right)^T\right)_{ij} = \begin{cases} M_{ij} - M_{(i-s),(j-s)} & \text{if } i > 1 \text{ and } j > 1 \\ M_{ij} & \text{otherwise} \end{cases}$$

Thus we can construct M uniquely from $+s(M)$ by first filling in the first s rows and columns of M with the same values as in $+s(M)$, and iteratively constructing the rest via

$$M_{ij} = +s\left(M\right)_{ij} + M_{(i-s),(j-s)}.$$

This means that the solution to $+s(M^{(+)}) = \sum_{\hat{t}} X^{(\hat{t})}(Y^{(\hat{t})})^T$ is unique, and all we have to do is to show that the formula we provided gives equality.

First, we express the $\{i, j\}$ block of the product of a lower triangular Toeplitz matrix and an upper triangular Toeplitz matrix can be in terms of the original blocks:

$$\left(T_L\left(X\right) T_L\left(Y\right)^T\right)_{\{i,j\}} = \sum_{k} T_L\left(X\right)_{\{i,k\}} T_L\left(Y\right)^T_{\{j,k\}} = \sum_{1 \leqslant k \leqslant \min\{i,j\}} X_{\{i+1-k,1\}} Y^T_{\{j+1-k,1\}}$$

Now consider the matrix with each row shifted down by 1 that results from multiplying by $\Delta(s)$. This leads to

$$\left(\Delta\left(s\right) T_L\left(X\right) T_L\left(X\right)^T \Delta\left(s\right)^T\right) = \left(\Delta\left(s\right) T_L\left(X\right)\right)\left(\Delta\left(s\right) T_L\left(Y\right)\right)^T.$$

Which when substituted into the above formula gives:

$$\left(\Delta\left(s\right) T_L\left(X\right) T_L\left(X\right)^T \Delta\left(s\right)^T\right)_{\{i,j\}} = \sum_{1 \leqslant k \leqslant \min\{i,j\}-1} X_{\{i-k,1\}} Y^T_{\{j-k,1\}}.$$

Upon comparison, the only different term is $X_{\{i,1\}}Y_{\{j,1\}}^T$, which is precisely the corresponding block in XY^T. Note that the column indices are 1 in both of these because X and Y are both ms-by-s, so exactly one column block. $\square$

Efficient multiplications against $T_L(X)$ and $T_L(X)^T$ can in turn be realized (in an operator manner) via fast Fourier transforms.

First, observe that for an ms-by-k matrix B, $T_L(X)B$ is an n-by-s matrix. If we interpret it as m s-by-s blocks, these blocks the result of computing the convolution of $X_{\{1,:\}} \ldots X_{\{m,:\}}$ with $B_{\{m,:\}} \ldots B_{\{1,:\}}$.

Raising m to a power of 2, and filling the rest of the blocks with 0s means it suffices to show that we can generate a covolution operator with good bit complexity. Due to the connection with Fourier transforms, it is more convenient for us to use 0-indexing of the blocks.

Lemma B.4. *Given t that is a power of 2, along with a length t sequence of s-by-s matrices $X^{(0)} \ldots X^{(t-1)}$, and any error δ, there is an algorithm corresponding to a linear operator $\widetilde{Z}_{Conv(X,\delta)}$ with at most $O(\log t \log(ts(1+\|\|X\|\|_\infty)/\delta))$ words after the decimal point such that for the exact convolution matrix $\overline{Z}_{Conv(X)}$ given by:*

$$\overline{Z}_{Conv(X)\{i,j\}} = X^{((i-j) \mod m)}$$

we have

$$\left\|\widetilde{Z}_{Conv(X,\delta)} - \overline{Z}_{Conv(X)}\right\|_F \leqslant \delta$$

and for any length t sequence of s-by-k matrices $B^{(0)} \ldots B^{(t-1)} \in \Re^{s\times k}$ with at most L_B words after decimal point, corresponding vertically concatenated matrix $B \in \Re^{ts\times k}$, evaluating $\widetilde{Z}_{Conv(X,\delta)}B$ takes time

$$O\left(t \log^2 t \cdot \max\left\{s^2 k^{\omega-2}, s^{\omega-1}k\right\} \cdot \left(\log\left(\left(1+\|\|X\|\|_\infty\right)\left(1+\|\|B\|\|_\infty\right) ts/\delta\right) + L_B\right)\right)$$

in the unit-cost RAM model.

Applying this convolution twice to the lower/upper block triangular matrices then gives the overall algorithm.

Proof. (of Lemma 6.6) Applying Lemma B.4 to the blocks of X and Y for some error $\widehat{\delta}$ that we will set at the end of this proof gives that there are routines for multiplying by $T_L(X)$ and $T_L(Y)^T$ that correspond to linear operators such that:

1. The error in Fronbenius norm is at most $\widehat{\delta}$.
2. The number of digits after decimal point is at most $O(\log m \log(ms(1+\|\|X\|\|_\infty)(1+\|\|Y\|\|_\infty)/\widehat{\delta}))$
3. For am ms-by-k matrix B with L_B digits after the decimal point, the evaluation cost for these operators is $O(m\log^2 m \max\{s^2k^\omega(\log((1+\|\|X\|\|_\infty)(1+\|\|Y\|\|_\infty)(1+\|\|B\|\|_\infty)ms/\widehat{\delta}) + L_B))$.

We then composing these operators for $L_T(X^{(\hat{t})})$ and $L_T(Y^{(\hat{t})})^T$ via the composition statement from Lemma 3.2. Note that we have

$$\left\|\left\| T_L\left(X\right)\right\|\right\|_{\infty} \leqslant \left\|\left\| X\right\|\right\|_{\infty}$$

as $T_L()$ simply duplicates the blocks in X. So it suffices to choose

$$\widehat{\delta} \leftarrow \frac{\delta}{ms\left(1+\left\|\left\|X\right\|\right\|_{\infty}\right)\left(1+\left\|\left\|Y\right\|\right\|_{\infty}\right)}.$$

Substituting this value into the bit-length of the operators gives that the number of bits in them is still $O(\log m \log(ms(1+\|\|X\|\|_{\infty})(1+\|\|Y\|\|_{\infty})/\delta))$, which in turn gives that the maximum number of words after decimal point passed as input when invoking them, for a particular B is at most

$$O\left(\log t \log\left(ms\left(1+\left\|\left\|X\right\|\right\|_{\infty}\right)\left(1+\left\|\left\|Y\right\|\right\|_{\infty}\right)/\delta\right)+L_B\right).$$

Substituting this into the algorithmic costs of Lemma B.4 along with the $\lceil r/s\rceil$ involved gives a total cost of

$$\begin{aligned}&\left\lceil\frac{r}{s}\right\rceil \cdot O\left(m\log^3 m\cdot \max\left\{s^2k^{\omega-2}, s^{\omega-1}k\right\}\right.\\ &\qquad \cdot\left(\log\left(\left(1+\left\|\left\|X\right\|\right\|_{\infty}\right)\left(1+\left\|\left\|Y\right\|\right\|_{\infty}\right)\left(1+\left\|\left\|B\right\|\right\|_{\infty}\right)ms/\delta\right)+L_B\right)\right)\\ &= O\left(m\log^3 m\cdot \max\left\{r,s\right\}\max\left\{sk^{\omega-2}, s^{\omega-2}k\right\}\right.\\ &\qquad \cdot\left(\log\left(\left(1+\left\|\left\|X\right\|\right\|_{\infty}\right)\left(1+\left\|\left\|Y\right\|\right\|_{\infty}\right)\left(1+\left\|\left\|B\right\|\right\|_{\infty}\right)ms/\delta\right)+L_B\right)\right)\end{aligned}$$

□

C Convolution Operator

In this section we bound the word complexity of the operator version of fast convolution.

Lemma B.4. *Given t that is a power of 2, along with a length t sequence of s-by-s matrices $X^{(0)}\ldots X^{(t-1)}$, and any error δ, there is an algorithm corresponding to a linear operator $\widetilde{Z}_{Conv(X,\delta)}$ with at most $O(\log t\log(ts(1+\|\|X\|\|_{\infty})/\delta))$ words after the decimal point such that for the exact convolution matrix $\overline{Z}_{Conv(X)}$ given by:*

$$\overline{Z}_{Conv(X)\{i,j\}} = X^{((i-j) \mod m)}$$

we have

$$\left\|\widetilde{Z}_{Conv(X,\delta)} - \overline{Z}_{Conv(X)}\right\|_F \leqslant \delta$$

and for any length t sequence of s-by-k matrices $B^{(0)}\ldots B^{(t-1)} \in \Re^{s\times k}$ with at most L_B words after decimal point, corresponding vertically concatenated matrix $B \in \Re^{ts\times k}$, evaluating $\widetilde{Z}_{Conv(X,\delta)}B$ takes time

$$O\left(t\log^2 t\cdot \max\left\{s^2k^{\omega-2}, s^{\omega-1}k\right\}\cdot\left(\log\left(\left(1+\left\|\left\|X\right\|\right\|_{\infty}\right)\left(1+\left\|\left\|B\right\|\right\|_{\infty}\right)ts/\delta\right)+L_B\right)\right)$$

in the unit-cost RAM model.

Note that all numbers in X can be truncated to $O(\log(t/\epsilon))$ words after the decimal point. After this, the easiest way of doing this is to treat the numbers as integers, and use modulus against primes (of size around t) to reduce this to computing convolutions over finite fields via the exact same formulas below (with ω replaced by a generator over said finite fields). Due to the involvement of different number systems, we omit details on such an implementation in favor of more thorough expositions, e.g. Chapters 30 and 31 of the Third Edition of Introduction to Algorithms by Cormen-Leiserson-Rivest-Stein [CLRS09].

Below we also verify how to do this using decimal expansions. We compute this convolution via real-valued Fast Fourier Transform (FFT) with all numbers rounded to $O(1/\epsilon)$ words of accuracy. We first briefly describe how the exact FFT algorithm works: a detailed explanation can be found in Chapter 30 [CLRS09].

Let

$$\omega = \cos\left(\frac{2\pi}{t}\right) + i \cdot \sin\left(\frac{2\pi}{t}\right)$$

be a t^{th} roof of unity. Then the i^{th} block of the Fourier transform of X is given by

$$\sum_j \omega^{i \cdot j} X^{(j)}$$

the corresponding block for B is given by:

$$\sum_k \omega^{i \cdot k} B^{(k)}.$$

Taking the product of these two blocks (with the same index i) then leads to:

$$\sum_{j,k} \omega^{i \cdot (j+k)} X^{(j)} B^{(k)}.$$

Note that the corresponding blocks have dimensions s-by-s and s-by-k respectively, so their product gives an s-by-k block.

Applying inverse FFT to these then gives blocks of the form (for index l):

$$\sum_{i,j,k} \omega^{-il} \cdot \omega^{i \cdot (j+k)} X^{(j)} B^{(k)} = \left[\sum_i \omega^{i \cdot (j+k-l)}\right] \cdot \sum_{j,k} X^{(j)} B^{(k)}.$$

The properties of roots of unity then gives that this leading coefficient is t if $j + k \equiv l \pmod{t}$, and 0 otherwise. So the above equations, when implemented exactly, gives t times the convolution between X and B.

We first sketch how the Fast Fourier Transform can be efficiently implemented as an linear operator in the presence of roundoff errors. Note that notions of Frobenius/ℓ_2 norms extend to the complex setting as well.

Lemma C.1. *Given any value t and block size s, and any $0 < \delta < 0.1$, there is an algorithm that correspond to a linear operator $\widetilde{Z}_{FFT(t,s,\delta)}$ with at most $O(\log(t/\delta)\log t)$ digits after the decimal place such that*

$$\left\| \widetilde{Z}_{FFT(t,s,\delta)} - \overline{Z}_{FFT(t,s)} \right\|_F \leqslant \delta,$$

where $\overline{Z}_{FFT(t,s)}$ is the exact FFT operator on t blocks of size s:

$$\left(\overline{Z}_{FFT(t,s)}\right)_{\{i,j\}} = \omega^{i \cdot j} \cdot I(s) \qquad \forall 1 \leqslant i, j \leqslant t,$$

where $I(s)$ is the s-by-s identity matrix. Given any matrix-sequence $X \in \Re^{ts \times k}$ with at most L_X words after the decimal place, $\widetilde{Z}_{FFT(t,s,\delta)}X$ can be computed in time $O(t \log t \cdot sk \cdot (1 + \log(\|\!|X|\!\|_\infty) + L_X + \log t \log(t/\delta)))$.

A similar bound also holds for the inverse FFT operator, $\widetilde{Z}_{InvFFT(t,s)}$, specifically $\left\| \widetilde{Z}_{InvFFT(t,s,\delta)} - Z_{InvFFT(t,s)} \right\|_F \leqslant \delta$ with the same running time / word length bounds.

Proof. We introduce round-off errors into the Fast Fourier transform algorithm, but only in one place: the generation of each complex coefficient used to evaluate the butterfly diagram.

Because all coefficients in the FFT matrix have magnitude at most 1, errors to the corresponding coefficients accumulate additively. So it suffices to have round-off error at most $\delta/poly(t)$ these coefficients.

This implies at most $O(\log(t/\delta))$ words after decimal point for each of the coefficients. As the depth of the FFT recursion is $O(\log t)$, the total length of the words are bounded by $O(\log t \log(t/\delta))$. This in turn implies that any linear combination of these coefficients and X can have at most $O(L_X + \log t \log(t/\delta)))$ words after the decimal place. Combining this with none of the intermediate numbers exceeding $t \|\!|X|\!\|_\infty$, and the $O(sk)$ cost of adding matrices gives the overall cost. $\square$

Proof. (Of Lemma B.4) We first round all entries in X to an additive accuracy of $\delta/(ts)$, or $O(\log(ts/\delta))$ words after the decimal point. This perturbs the convolution operator in X by at most δ/ts.

We apply the FFT operators to X and B separately, namely we first invoke the FFT algorithm from Lemma C.1 to compute $\widetilde{Z}_{FFT(t,s,\widehat{\delta})}X$ and $\widetilde{Z}_{FFT(t,s,\widehat{\delta})}B$ for some $\widehat{\delta}$ that we will choose later as a function of δ, $\|\!|X|\!\|_\infty$, $\|\!|B|\!\|_\infty$, and ms.

The cost of these two (forward) FFT transforms is then at most

$$\begin{aligned} O&\left(t \log t \cdot s^2 \cdot (\log(1 + \|\!|X|\!\|_\infty) + \log(ts/\delta)) + \log t \log\left(t/\widehat{\delta}\right)\right) \\ &\qquad + O\left(t \log t \cdot sk \cdot \left(\log(1 + \|\!|B|\!\|_\infty) + L_B + \log t \log\left(t/\widehat{\delta}\right)\right)\right) \\ \leqslant O&\left(t \log t \cdot (s^2 + sk) \cdot \left(\log((1 + \|\!|X|\!\|_\infty)(1 + \|\!|B|\!\|_\infty) ms/\delta) + L_B + \log t \log\left(t/\widehat{\delta}\right)\right)\right) \end{aligned}$$

The products of the corresponding s-by-s and s-by-k matrices involve numbers with at most $L_X + O(\log(t/\widehat{\delta}) \log t) = O(\log(t/\widehat{\delta}) \log t)$ words and $L_B + O(\log(t/\widehat{\delta}) \log t)$ words after the decimal point respectively. So the computation of the product involves numbers of length at most

$$O\left(\log\left(\left(1+\left\|\left|X\right\|\right|_{\infty}\right)\left(1+\left\|\left|B\right\|\right|_{\infty}\right)\right) + L_B + \log(t/\widehat{\delta}) \log t\right).$$

When $k \leqslant s$, it can be reduced to multiplying $\lceil s/k \rceil^2$ k-by-k matrices, for a total of $O((s/k)^2 k^{\omega}) = O(s^2 k^{\omega-2})$. When $k \geqslant s$, it reduces to $\lceil k/s \rceil$ multiplications of s-by-s matrices, for a total of $O((k/s)s^{\omega}) = O(ks^{\omega-1})$. As the larger exponent is only the larger term, the overall running time is bounded by the max of these two.

The i^{th} block of this matrix product

$$\left[\widetilde{Z}_{FFT\left(t,s,\widehat{\delta}\right)} X\right]_{\{i\}} \cdot \left[\widetilde{Z}_{FFT\left(t,s,\widehat{\delta}\right)} B\right]_{\{i\}}.$$

can also be viewed as a linear transformation of B, namely the matrix

$$\widetilde{Z}_{FFT\left(t,s,\widehat{\delta}\right),\{i\}} X \widetilde{Z}_{FFT\left(t,s,\widehat{\delta}\right),\{i\}}.$$

Applying the error composition bound given in Lemma 3.2 gives that its error to the exact operator is bounded by

$$\left\|\widetilde{Z}_{FFT\left(t,s,\widehat{\delta}\right),\{i\}} X \widetilde{Z}_{FFT\left(t,s,\widehat{\delta}\right),\{i\}} - \overline{Z}_{FFT\left(t,s,\widehat{\delta}\right),\{i\}} X \overline{Z}_{FFT\left(t,s,\widehat{\delta}\right),\{i\}}\right\|_F$$
$$\leqslant 100\widehat{\delta} \max\left\{1, \left\|X\right\|_2\right\} \leqslant 100\widehat{\delta}\,(ts)^2\left(1+\left\|\left|X\right\|\right|_{\infty}\right).$$

Summing over all $0 \leqslant i < t$ gives a total error that's larger by a factor of (ts).

Composing this matrix once more against the inverse FFT given by Lemma C.1 then gives the overall error bound. Specifically, for an overall operator error of δ, it suffices to set

$$\widehat{\delta} \leftarrow \frac{\delta}{\left(1+\left\|\left|X\right\|\right|_{\infty}\right) t^2 s^2}.$$

This means each FFT/inverse FFT increases the number of digits behind decimal place by at most

$$O\left(\log t \log\left(t/\widehat{\delta}\right)\right) = O\left(\log t \log\left(\left(1+\left\|\left|X\right\|\right|_{\infty}\right) ts/\delta\right)\right).$$

additively, which in turn gives the bound on the number of digits after decimal place for $\widetilde{Z}_{FFT(t,s,\delta)}$.

Incorporating the size of B, we get that the total number of words after decimal point that we need to track is $O(L_B + \log t \log((1+\left\|\left|X\right\|\right|_{\infty})ts/\delta))$ On the other hand, none of the intermediate matrices during the convolution have magnitude more than $(ts)^2 \left\|\left|X\right\|\right|_{\infty} \left\|\left|B\right\|\right|_{\infty}$. This means the total number of words tracked in order to represent all intermediate results is at most

$$O\left(\log\left(\left(1+\left\|\left|X\right\|\right|_{\infty}\right)\left(1+\left\|\left|B\right\|\right|_{\infty}\right) ts\right) + L_B + \log t \log\left(\left(1+\left\|\left|X\right\|\right|_{\infty}\right) ts/\delta\right)\right).$$

So combining this with the cost of matrix multiplication, and the $O(t \log t)$ block rearrangements of the FFT steps gives a total cost upper bound of

$$O\left(t \log^2 t \cdot \max\left\{s^2 k^{\omega-2}, s^{\omega-1} k\right\} \cdot \left(\log\left(\left(1 + \|X\|_\infty\right)\left(1 + \|B\|_\infty\right) ts/\delta\right) + L_B\right)\right)$$

word operations. $\square$

Notation

A	matrix to be solved
b, B	right-hand side vector(s) of linear equations
n	size of matrix
m	number of Krylov space steps
s	size of Krylov space blocks
r	number of remaining columns 'padded' to block-Krylov space, also number of columns of null space basis we work with.
K	block-Krylov space
$\textsc{Solve}_A(\cdot)$	solve procedure for matrix A that takes as input b, and outputs $x \approx A^{-1}b$
$Z_{\textsc{Alg}}$	linear operator corresponding to $\textsc{Alg}$
Q	full n-by-n matrix formed by appending columns a Krylov space matrix
g	Gaussian vector
G	Gaussian matrix
g^S, G^S	sparse vector/matrix with non-zeros set to Gaussians
$[\cdot]$	subset of m columns in Krylov space corresponding to a single initial vector.
$\{\cdot\}$	subset of s columns in Krylov space corresponding to a particular power.
I	identity matrix
U, W	orthonormal basis
d	dimension of a basis
u, w	unit vectors
λ	eigenvalues
v	eigenvectors
Λ	diagonal matrix containing eigenvalues
σ	singular values
Σ	diagonal matrix containing singular values
κ	condition number
ϵ	granularity of ϵ-nets
α	non-degeneracy size
$C, \overline{C}$	remaining/eliminated split of coordinates in block Gaussian elimination
X, Y	matrices that form low rank approximations, and tall-and-thin matrices
Z	matrices/linear operators that correspond to linear algorithms
$\cdot_S$, $\cdot_J$	subset of columns of matrices
$\left\Vert\cdot\right\Vert_2$	2-norm of vectors
$\left\Vert\cdot\right\Vert_F$	Frobenius norms of matrices
$\left\vert\kern-0.25ex\left\vert\kern-0.25ex\left\vert\cdot\right\vert\kern-0.25ex\right\vert\kern-0.25ex\right\vert_\infty$	entry-wise max magnitude of matrices
$A-$	A with last row/column removed (only in Lemma 4.1)
θ	rescaling/renormalizing coefficients

Table 1: Variable Names and Meaning