Sublinear Time Low-Rank Approximation of Positive Semidefinite Matrices

Cameron Musco
Massachusetts Institute of Technology
Cambridge, MA
cnmusco@mit.edu

David P. Woodruff
Carnegie Mellon University
Pittsburgh, PA
dwoodruf@cs.cmu.edu

***Abstract*—We show how to compute a *relative-error* low-rank approximation to any positive semidefinite (PSD) matrix in *sublinear time*, i.e., for any $n \times n$ PSD matrix A, in $\tilde{O}(n \cdot \mathrm{poly}(k/\epsilon))$ time we output a rank-k matrix B, in factored form, for which $\|A - B\|_F^2 \leq (1+\epsilon)\|A - A_k\|_F^2$, where A_k is the best rank-k approximation to A. When k and $1/\epsilon$ are not too large compared to the sparsity of A, our algorithm does not need to read all entries of the matrix. Hence, we significantly improve upon previous $\mathrm{nnz}(A)$ time algorithms based on oblivious subspace embeddings, and bypass an $\mathrm{nnz}(A)$ time lower bound for general matrices (where $\mathrm{nnz}(A)$ denotes the number of non-zero entries in the matrix). We prove time lower bounds for low-rank approximation of PSD matrices, showing that our algorithm is close to optimal. Finally, we extend our techniques to give sublinear time algorithms for low-rank approximation of A in the (often stronger) spectral norm metric $\|A - B\|_2^2$ and for ridge regression on PSD matrices.**

***Keywords*-low-rank approximation; leverage score sampling; sublinear time algorithms; matrix sketching**

For full paper version see `https://arxiv.org/abs/1704.03371`.

I. Introduction

A fundamental task in numerical linear algebra is to compute a low-rank approximation of a matrix. Such an approximation can reveal underlying low-dimensional structure, can provide a compact way of storing a matrix in factored form, and can be quickly applied to a vector. Countless applications include clustering [1]–[4], datamining [5], information retrieval [6], learning mixtures of distributions [7], [8], recommendation systems [9], topic modeling [10], and web search [11], [12].

One of the most well-studied versions of the problem is to compute a near optimal low-rank approximation with respect to the Frobenius norm. That is, given an $n \times n$ input matrix A and an accuracy parameter $\epsilon > 0$, output a rank-k matrix B for which:

$$\|A - B\|_F^2 \leq (1+\epsilon)\|A - A_k\|_F^2, \tag{1}$$

where for a matrix C, $\|C\|_F^2 = \sum_{i,j} C_{i,j}^2$ is its squared Frobenius norm, and $A_k = \mathrm{argmin}_{\text{rank-}k\ B}\|A - B\|_F$. A_k can be computed exactly using the singular value decomposition, but takes $O(n^3)$ time in practice and n^ω time in theory, where $\omega \approx 2.373$ is the exponent of matrix multiplication.

In seminal work, Frieze, Kannan, and Vempala [13] and Achlioptas and McSherry [14] show that using randomization and approximation, much faster runtimes are possible. Specifically, [13] gives an algorithm that, assuming access to the row norms of A, outputs rank-k B, in factored form, such that with good probability, $\|A - B\|_F^2 \leq \|A - A_k\|_F^2 + \epsilon\|A\|_F^2$. The algorithm runs in just $n \cdot \mathrm{poly}(k/\epsilon)$ time. However $\mathrm{nnz}(A)$ additional time is required to compute the row norms, where $\mathrm{nnz}(A)$ denotes the number of non-zero entries of A. Further, the guarantee achieved can be significantly weaker than (1), since the error is of the form $\epsilon\|A\|_F^2$ rather than $\epsilon\|A - A_k\|_F^2$. Note that $\|A - A_k\|_F^2 \ll \|A\|_F^2$ precisely when A is well-approximated by a rank-k matrix. Related additive error algorithms with additional assumptions were given for tensors in [15].

Sarlós [16] showed how to achieve (1) with constant probability in $\tilde{O}(\mathrm{nnz}(A) \cdot k/\epsilon) + n \cdot \mathrm{poly}(k/\epsilon)$ time. This was improved by Clarkson and Woodruff [17] who achieved $O(\mathrm{nnz}(A)) + n \cdot \mathrm{poly}(k/\epsilon)$ time. See also work by Bourgain, Dirksen, and Nelson [18], Cohen [19], Meng and Mahoney [20], and Nelson and Nguyen [21] which further improved the degree in the $\mathrm{poly}(k/\epsilon)$ term. For a survey, see [22].

In the special case that A is rank-k and so $\|A - A_k\|_F^2 = 0$, (1) is equivalent to the well studied low-rank matrix completion problem [23]. Much attention has focused on completing *incoherent* low-rank matrices, whose singular directions are represented uniformly throughout the rows and columns and hence can be identified via uniform sampling and without fully accessing the matrix. Under incoherence assumptions, a number of methods are able to complete a rank-k matrix in $\tilde{O}(n \cdot \mathrm{poly}(k))$ time [24], [25].

For general matrices, without incoherence, it is not hard to see that $\Omega(\mathrm{nnz}(A))$ is a time lower bound: if one does not read a constant fraction of entries of A, with constant probability one can miss an entry much larger than all others, which needs to be included in the low-rank approximation.

A. Low-rank Approximation of PSD Matrices

An important class of matrices for which low-rank approximation is often applied is the set of positive semidefinite (PSD) matrices. These are real symmetric matrices with all non-negative eigenvalues. They arise for example as covariance matrices, graph Laplacians, Gram matrices (in particular, kernel matrices), and random dot product models [26]. In multidimensional scaling, low-rank approximation of PSD matrices in the Frobenius norm error metric (1) corresponds to the standard 'strain minimization' problem [27]. Completion of low-rank, or nearly low-rank (i.e., when $\|A-A_k\|_F^2 \approx 0$), PSD matrices from few entries is important in applications such as quantum state tomography [28] and global positioning using local distances [29], [30].

Due to its importance, a vast literature studies low-rank approximation of PSD matrices [31]–[43]. However, known algorithms either run in at least $\mathrm{nnz}(A)$ time, do not achieve the relative-error guarantee of (1), or require strong incoherence assumptions.[1]

At the same time, the simple $\Omega(\mathrm{nnz}(A))$ time lower bound for general matrices *does not hold in the PSD case*. Positive semidefiniteness ensures that for all i, j, $|A_{i,j}| \leq \max(A_{i,i}, A_{j,j})$. So 'hiding' a large entry in A requires creating a corresponding large diagonal entry. By reading the n diagonal elements, an algorithm can avoid being tricked by this approach. While far from an algorithm, this argument raises the possibility that improved runtimes could be possible for PSD matrices.

B. Our Results

We give the first sublinear time relative-error low-rank approximation algorithm for PSD matrices. Our algorithm reads just $nk \cdot \mathrm{poly}(\log n/\epsilon)$ entries of A and runs in $nk^{\omega-1} \cdot \mathrm{poly}(\log n/\epsilon)$ time (Theorem 9). With probability $99/100$ it outputs a matrix B in factored form which satisfies (1). We critically exploit the intuition that large entries cannot 'hide' in PSD matrices, but surprisingly require *no additional assumptions* on A, such as incoherence or bounded condition number.

We complement our algorithm with an $\Omega(nk/\epsilon)$ time lower bound. The lower bound is information-theoretic, showing that any algorithm which reads fewer than this number of entries in the input cannot achieve the guarantee of (1) with constant probability. As our algorithm only reads $nk \cdot \mathrm{poly}(\log n/\epsilon)$ entries of A, this is nearly optimal for constant ϵ. We note that the actual time complexity of our algorithm is slower by a factor of $k^{\omega-2}$.

Finally, we show that our techniques can be extended to compute B satisfying the spectral norm guarantee: $\|A - B\|_2^2 \leq (1+\epsilon)\|A-A_k\|_2^2 + \frac{\epsilon}{k}\|A-A_k\|_F^2$ using just $nk^2 \cdot \mathrm{poly}(\log n/\epsilon)$ accesses to A and $nk^\omega \cdot \mathrm{poly}(\log n/\epsilon)$ time (Theorem 18). This guarantee is often stronger than (1) when $\|A-A_k\|_F^2$ is large, and is important in many applications. For example, we use this result to solve the ridge regression problem $\min_{x\in\mathbb{R}^n} \|Ax-y\|_2^2 + \lambda\|x\|_2^2$ up to $(1+\epsilon)$ relative error in $\tilde{O}\left(\frac{ns_\lambda^\omega}{\epsilon^{2\omega}}\right)$ time, where $s_\lambda = \mathrm{tr}((A^2+\lambda I)^{-1}A^2)$ is the *statistical dimension* of the problem (see Theorem 19). Typically $s_\lambda \ll n$, so our runtime is sublinear and improves significantly on existing input-sparsity time results [44]. For a summary of our results and comparison to prior work, see Table 1 of our full paper.

C. Algorithm Overview

The starting point for our approach is the fundamental fact that *any* matrix A contains a subset of $O(k/\epsilon)$ columns, call them C, that span a relative-error rank-k approximation to A [45]–[47]. Computing the best low-rank approximation to A using an SVD requires access to all $\Theta(n^2)$ dot products between the columns of the matrix. However, given C, just $n \cdot O(k/\epsilon)$ dot products are needed – to project the remaining columns of the matrix to the span of the subset.

Additionally, a subset of size $\mathrm{poly}(k/\epsilon)$ can be identified using an intuitive approach known as *adaptive sampling* [46]: columns are iteratively added to the subset, with each new column being sampled with probability proportional to its norm *outside the column span* of the current subset. Formally, column a_i is selected with probability $\frac{\|a_i - P_C a_i\|_2^2}{\|A-P_C A\|_F^2}$ where P_C is the projection onto the current subset C. Computing these sampling probabilities requires knowing the norm of each a_i along with its dot product with each column currently in C. So, overall this approach gives a relative-error low-rank approximation using just $n \cdot \mathrm{poly}(k/\epsilon)$ dot products between columns of A.

The above observation is surprising – not only does every matrix contain a small column subset witnessing a near optimal low-rank approximation, but also, such a witness can be found using significantly less information about the column span of the matrix than is required by a full SVD.

This fact is not immediately algorithmically useful, as computing the required dot products takes $\mathrm{nnz}(A) \cdot \mathrm{poly}(k/\epsilon)$ time. However, given PSD A, we can write the eigendecomposition $A = U\Lambda U^T$ where Λ is a non-negative diagonal matrix of eigenvalues, and let $A^{1/2} = U\Lambda^{1/2}U^T$ be the matrix square root of A. Since $A^{1/2}A^{1/2} = A$, the entry $A_{i,j}$ is just the dot product between the i^{th} and j^{th} columns of $A^{1/2}$. So with A in hand, the dot products have been 'precomputed' and the above approach yields a low-rank approximation algorithm for $A^{1/2}$ running in just $n \cdot \mathrm{poly}(k/\epsilon)$ time. Note that, aligning with our initial intuition that reading the diagonal entries of A is necessary to avoid the $\mathrm{nnz}(A)$ time lower bound for general matrices, the diagonal entries of A are the column norms of $A^{1/2}$, and hence their values are critical to computing the adaptive sampling probabilities.

By the above argument, given PSD A, we can compute in $n \cdot \mathrm{poly}(k/\epsilon)$ time a rank-k orthogonal projection matrix $P \in \mathbb{R}^{n\times n}$ (in factored form) for which $\|A^{1/2} - A^{1/2}P\|_F^2 \leq (1+\epsilon)\|A^{1/2} - A_k^{1/2}\|_F^2$. This approach can be implemented using adaptive sampling [46], sublinear time volume sampling [48], or as shown in [42], recursive *ridge leverage score sampling*. The ridge leverage scores are a natural interpolation between adaptive sampling and the widely studied

[1] Many algorithms satisfy the additional constraint that the low-rank approximation B is PSD. This is also known to be possible in $O(\mathrm{nnz}(A))$ time using sketching-based algorithms for general matrices [43].

leverage scores, which, as we will see, have a number of additional algorithmically useful properties. As discussed in [42], the guarantee for $A^{1/2}$ is useful for a number of kernel learning methods such as kernel ridge regression. However, it is very different from our final goal. In fact, one can show that projecting to P can yield an *arbitrarily bad* low-rank approximation to A itself (see Appendix A in our full paper).

We note that, since P is constructed via column selection methods, it is possible to efficiently compute a factorization of $A^{1/2}PA^{1/2}$ (see Appendix A in our full paper). Further, this matrix gives a near optimal low-rank approximation of A if we use error $\epsilon' = \epsilon/\sqrt{n}$. This gives a first sublinear time algorithm, but it is significantly suboptimal. Namely, it requires reading $\tilde{O}(nk/\epsilon') = \tilde{O}(n^{3/2}k/\epsilon)$ entries of A and takes $n^{1.69} \cdot \operatorname{poly}(k/\epsilon)$ time using fast matrix multiplication.

To improve the dependence on n, we need a better understanding of how to perform ridge leverage score sampling on A itself. We start by showing that the ridge leverage scores of $A^{1/2}$ are within a factor of $O(\sqrt{n/k})$ of the ridge leverage scores of A. By this bound, if we over-sample columns of A by a factor of $O(\sqrt{n/k})$ using the ridge leverage scores of $A^{1/2}$ (computable via [42]), obtaining a sample of $\tilde{O}(\sqrt{n/k} \cdot k/\epsilon^2)$ columns, the sample will be a so-called *projection-cost preserving sketch* (PCP) of A. The notion of a PCP was introduced in [4]: C is an (ϵ, k)-column PCP of A if for all rank-k projection matrices P,

$$(1-\epsilon)\|A - PA\|_F^2 \leq \|C - PC\|_F^2 \leq (1+\epsilon)\|A - PA\|_F^2. \quad (2)$$

One important property of a PCP is that good low-rank approximations to C translate to good low-rank approximations of A. More precisely, if U is an $n \times k$ matrix with orthonormal columns for which $\|C - UU^TC\|_F^2 \leq (1+\epsilon)\|C - C_k\|_F^2$, then $\|A - UU^TA\|_F^2 \leq \frac{(1+\epsilon)^2}{(1-\epsilon)}\|A - A_k\|_F^2$.

Letting C be the $n \times \tilde{O}(\sqrt{nk}/\epsilon^2)$ submatrix which we sample via ridge leverage scores, we can apply an $\operatorname{nnz}(C)$ time algorithm to compute $U \in \mathbb{R}^{n\times k}$ whose columns span a near-optimal low-rank approximation of C, and hence of A by the PCP property. Using standard sampling techniques, we can approximately project the columns of A to U, producing our final solution. This gives time complexity $n^{3/2} \cdot \operatorname{poly}(k/\epsilon)$, slightly improving upon our first approach.

To reduce the time to linear in n, we must further reduce the size of C by sampling a subset of its rows, which themselves form a PCP. To find these rows, we cannot use sketching techniques, which would take at least $\operatorname{nnz}(C)$ time, nor can we use our previous method for providing $O(\sqrt{n/k})$ overestimates to the ridge leverage scores, since C is no longer PSD. In fact, the row ridge leverage scores of C can be arbitrarily large compared to those of $A^{1/2}$.

The key idea to getting around this issue is that, since C is a column PCP of A, projecting its columns onto A's top eigenvectors gives a near optimal low-rank approximation. Further, we can show that the ridge leverage scores of $A^{1/2}$ (appropriately scaled) upper bound the *standard leverage scores* of this low-rank approximation. Sampling by these leverage scores is not enough to give a guarantee like (2) – they ignore the entire component of C not falling in the span of A's top eigenvectors and so may significantly distort projection costs over the matrix. Further, it is unclear how to estimate the row norms of C, or even its Frobenius norm, with $n \operatorname{poly}(k/\epsilon)$ samples, which are necessary to implement any kind of adaptive sampling approach.

Fortunately, using that row sampling at least preserves C in expectation, along with a few other properties of the ridge leverage scores of $A^{1/2}$, we show that, with good probability, sampling $\tilde{O}(\sqrt{nk}/\operatorname{poly}(\epsilon))$ rows of C by these scores yields R satisfying for all rank-k projection matrices P:

$$(1-\epsilon)\|C - CP\|_F^2 \leq \|R - RP\|_F^2 + \Delta \leq (1+\epsilon)\|C - CP\|_F^2$$

where Δ is a fixed value, independent of P, with $|\Delta| \leq c\|C - C_k\|_F^2$ for some constant c. Since the same Δ distortion applies to all P, and since it is at most a constant times the true optimum, a near optimal low-rank approximation for R still translates to a near optimal approximation for C.

At this point R is a small matrix, and we can run any $O(\operatorname{nnz}(R))$ time algorithm to find a good low-rank factorization EF^T to it, where F^T is $k \times \tilde{O}(\sqrt{nk}/\operatorname{poly}(\epsilon))$. Since R is a row PCP for C, by regressing the rows of C to the span of F, we can obtain a near optimal low-rank approximation to C. We can solve this multi-response regression approximately in sublinear time via standard sampling techniques. Approximately regressing A to the span of this approximation using similar techniques yields our final result. The total runtime is dominated by the input-sparsity low-rank approximation of R requiring $O(\operatorname{nnz}(R)) = \tilde{O}(nk/\operatorname{poly}(\epsilon))$ time.

To improve ϵ dependencies in our final runtime, achieving sample complexity $\tilde{O}\left(\frac{nk}{\epsilon^{2.5}}\right)$, we modify this approach somewhat, showing that R actually satisfies a stronger *spectral norm PCP* property for C. This property lets us find a low-rank span Z with $\|C - CZZ^T\|_2^2 \leq \frac{\epsilon}{k}\|A - A_k\|_F^2$, from which, through a series of approximate regression steps, we can extract a low-rank approximation to A satisfying (1). This stronger spectral guarantee also lies at the core of our extensions to near optimal spectral norm low-rank approximation (Theorem 18), ridge regression (Theorem 19), and low-rank approximation where B is restricted to be PSD (Theorem 10).

D. Some Further Intuition on Error Guarantees

Observe that in computing a low-rank approximation of A, we read just $\tilde{O}(n \cdot \operatorname{poly}(k/\epsilon))$ entries of the matrix, which is, up to lower order terms, the same number of entries (corresponding to column dot products of $A^{1/2}$) that we accessed to compute a low-rank approximation of $A^{1/2}$ in our description above. However, these sets of entries are very different. While low-rank approximation of $A^{1/2}$ looks at an $n \times \text{poly}(k/\epsilon)$ sized submatrix of A together with the diagonal entries, our algorithm considers a carefully chosen $\sqrt{nk}\,\text{poly}(\log n/\epsilon) \times \sqrt{nk}\,\text{poly}(\log n/\epsilon)$ submatrix together with the diagonal entries, which gives significantly more information about the spectrum of A.

As a simple example, consider A with top eigenvalue $\lambda_1 = \sqrt{n}$, and $\lambda_i = 1$ for $i = 2, ...n$. $\|A^{1/2}\|_F^2 = \sum_{i=1}^n \lambda_i = \sqrt{n}+n-1$ while $\|A^{1/2} - A_1^{1/2}\|_F^2 = \sum_{i=2}^n \lambda_i = n-1$. So, $A^{1/2}$ has no good rank-1 approximation. Unless we set $\epsilon = O(1/\sqrt{n})$, a low-rank approximation algorithm for $A^{1/2}$ can learn nothing about λ_1 and still be near optimal. In contrast, $\|A\|_F^2 = \sum_{i=1}^n \lambda_i^2 = 2n-1$ and $\|A - A_1\|_F^2 = \sum_{i=2}^n \lambda_i^2 = n-1$. So, even with $\epsilon = 1/2$, any rank-1 approximation algorithm for A must identify the presence of λ_1 and project this direction off the matrix. In this sense, our algorithm is able to obtain a much more accurate picture of A's spectrum.

With incoherence assumptions, prior work on PSD low-rank approximation [36] obtains the bound $\|A-B\|_* \leq (1+\epsilon)\|A-A_k\|_*$ in sublinear time, where $\|M\|_* = \sum_{i=1}^n \sigma_i(M)$ is the nuclear norm of M. Recent work ([48] in combination with [34]) gives $\|A - B\|_F \leq (k+1)\|A - A_k\|_*$ without the incoherence assumption. These nuclear norm bounds are closely related to approximation bounds for $A^{1/2}$ and it is not hard to see that neither require λ_1 to be detected in the example above, and so in this sense are weaker than our Frobenius norm bound.

A natural question if even stronger bounds are possible: e.g., can we compute B with $\|A - B\|_2^2 \leq (1+\epsilon)\|A - A_k\|_2^2$ in sublinear time? We partially answer this question in Theorem 18. In $\tilde{O}(nk^\omega \text{poly}(\log n/\epsilon))$ time, we can find B satisfying $\|A-B\|_2^2 \leq (1+\epsilon)\|A-A_k\|_2^2 + \frac{\epsilon}{k}\|A-A_k\|_F^2$.

Significantly improving the above bound seems hard: it is easy to see that a relative error spectral norm guarantee requires $\Omega(n^2)$ time. Consider A which is the identity except with $A_{i,j} = A_{j,i} = 1$ for some random pair (i,j). Finding (i,j) requires $\Omega(n^2)$ queries to A. However, it is necessary to achieve a relative error spectral norm guarantee with $\epsilon < 3$ since $\|A\|_2^2 = 4$ while $\|A - A_1\|_2^2 = 1$ where A_1 is all zeros with ones at its (i,i), (j,j), (i,j), and (j,i) entries.

A similar argument shows that relative error low-rank approximation in higher Schatten-p norms, i.e., $\|A - B\|_p^p$ for $p > 2$ requires superlinear dependence on n (where $\|M\|_p^p = \sum_{i=1}^n \sigma_i^p(M)$.) We can set A to be the identity but with an all ones block on a uniform random subset of $n^{1/p}$ indices. This block has associated eigenvalue $\lambda_1 = n^{1/p}$ and so, since all other $(n - n^{1/p})$ eigenvalues of A are 1, $\|A\|_p^p = \Theta(n)$, and the block must be recovered to give a relative error approximation to $\|A - A_1\|_p^p$. However, as the block is placed uniformly at random and contains just $n^{2/p}$ entries, finding even a single entry requires $n^{2-2/p}$ queries to A – superlinear for $p > 2$.

E. Open Questions

While it is apparent that obtaining stronger error guarantees than (1) may require increased runtime, understanding exactly what can be achieved in sublinear time is an interesting direction for future work. We also note that it is still unknown how to compute a number of basic properties of PSD matrices in sublinear time. For example, while we can output B satisfying $\|A - B\|_F^2 \leq (1+\epsilon)\|A - A_k\|_F^2$, surprisingly it is not clear how to actually estimate the value $\|A - A_k\|_F^2$ to within a $(1 \pm \epsilon)$ factor. This can be achieved in $n^{3/2}\text{poly}(k/\epsilon)$ time using our PCP techniques. However, obtaining linear runtime in n is open. Estimating $\|A - A_k\|_F^2$ seems strongly connected to estimating other important quantities such as the statistical dimension of A for ridge regression (see Theorem 19) which we do not know how to do in $o(n^{3/2})$ time.

Finally, an open question is if these techniques can be generalized to a broader class of matrices. As discussed, in the matrix completion literature, much attention has focused on incoherent low-rank matrices [23] which can be approximated with uniform sampling. PSD matrices are not incoherent in general, which is highlighted by the fact that our sampling schemes are far from uniform and very adaptive to previously seen matrix entries. However, perhaps there is some other parameter (maybe relating to a measure of diagonal dominance) which characterizes when low-rank approximation can be performed with just a small number of adaptive accesses to A.

F. Paper Outline

Section II: Ridge Leverage Score Sampling. We show that the ridge leverage scores of A are within an $O(\sqrt{n/k})$ factor of those of $A^{1/2}$, letting us use the fast ridge leverage score sampling algorithm of [42] to sample $\tilde{O}(\sqrt{nk}/\epsilon^2)$ columns of A that form a column PCP of the matrix.

Section III: Row Sampling. We discuss how to further accelerate our algorithm by obtaining a row PCP for our column sample, allowing us to achieve runtime linear in n.

Section IV: Full Algorithm. We use the primitives in the previous sections along with approximate regression techniques to give our full sublinear time low-rank approximation algorithm.

Section V: Lower Bounds. We show that our algorithm is nearly optimal – any relative error low-rank approximation algorithm must read $\Omega(nk/\epsilon)$ entries of A.

Section VI: Spectral Norm Bounds. We modify the algorithm of Section IV to give a tighter approximation in the spectral norm and discuss applications to ridge regression.

II. Ridge Leverage Score Sampling

Our main algorithmic tool will be ridge leverage score sampling, which is used to identify a small subset of columns of A that span a good low-rank approximation of the matrix. Following the definition of [49], the rank-k ridge leverage scores of any matrix A are given by:

Definition 1 (Ridge Leverage Scores). *For any $A \in \mathbb{R}^{n \times d}$, letting $a_i \in \mathbb{R}^n$ be the i^{th} column of A, the i^{th} rank-k column ridge leverage score of A is:*

$$\tau_i^k(A) = a_i^T \left(AA^T + \frac{\|A - A_k\|_F^2}{k} I \right)^+ a_i.$$

Above I is the appropriately sized identity matrix and M^+ denotes the matrix pseudoinverse, equivalent to the inverse unless $\|A - A_k\|_F^2 = 0$ and A is singular. Analogous scores can be defined for the rows of A by simply transposing the matrix. It is not hard to see that $0 < \tau_i^k(A) < 1$ for all i. Since we use these scores as sampling probabilities, it is critical that the sum of scores, and hence the size of the subsets we sample, is not too large. We have the following (see Appendix B in full paper):

Lemma 2 (Sum of Ridge Leverage Scores). *For any $A \in \mathbb{R}^{n \times d}$, $\sum_{i=1}^d \tau_i^k(A) \leq 2k$.*

Intuitively, the ridge leverage scores are similar to the standard leverage scores of A, which are given by $a_i^T(AA^T)^+ a_i$. By writing $A = U\Sigma V^T$ in its SVD, one sees that standard leverage scores are just the squared column norms of V^T. Sampling columns by ridge leverage scores yields a spectral approximation to the matrix. The addition of the weighted identity (or 'ridge') $\frac{\|A-A_k\|_F^2}{k} I$ 'dampens' contributions from smaller singular directions of A, decreasing the sum of the scores and allowing us to sample fewer columns. At the same time, it introduces error dependent on the size of the tail $\|A - A_k\|_F^2$, ultimately giving an approximation from which it is possible to output a near optimal low-rank approximation to the original matrix. Specifically, sampling by ridge leverage scores yields a *projection-cost preserving* sketch (PCP) of A:

Lemma 3 (Theorem 6 of [49]). *For any $A \in \mathbb{R}^{n \times d}$, for $i \in \{1, \dots, d\}$, let $\tilde{\tau}_i^k \geq \tau_i^k(A)$ be an overestimate for the i^{th} rank-k ridge leverage score. Let $p_i = \frac{\tilde{\tau}_i^k}{\sum_i \tilde{\tau}_i^k}$ and $t = \frac{c \log(k/\delta)}{\epsilon^2} \sum_i \tilde{\tau}_i^k$ for any $\epsilon < 1$ and sufficiently large constant c. Construct C by sampling t columns of A, each set to $\frac{1}{\sqrt{tp_i}} a_i$ with probability p_i. With probability $1 - \delta$, for any rank-k orthogonal projection $P \in \mathbb{R}^{n \times n}$,*

$$(1 - \epsilon)\|A - PA\|_F^2 \leq \|C - PC\|_F^2 \leq (1 + \epsilon)\|A - PA\|_F^2.$$

We refer to C as an (ϵ, k)-column PCP of A.

Since the 'cost' $\|A - PA\|_F^2$ of any rank-k projection of A is preserved by C, any near-optimal low-rank approximation of C yields a near optimal low-rank approximation of A. Further, C is much smaller than A, so such a low-rank approximation can be computed quickly. The difficulty is in computing the approximate leverage scores. To do this, we use the main result from [42]:

Lemma 4 (Corollary of Theorem 20 of [42]). *There is an algorithm that given any PSD matrix $A \in \mathbb{R}^{n \times n}$, runs in $O(n(k \log(k/\delta))^{\omega - 1})$ time, accesses $O(nk \log(k/\delta))$ entries of A, and returns for each $i \in [1, .., n]$, $\tilde{\tau}_i^k(A^{1/2})$ such that with probability $1 - \delta$, for all i:*

$$\tau_i^k(A^{1/2}) \leq \tilde{\tau}_i^k(A^{1/2}) \leq 3\tau_i^k(A^{1/2}).$$

Proof: Theorem 20 of [42] shows that by using a recursive ridge leverage score sampling algorithm, it is possible to return (with probability $1 - \delta$) a sampling matrix $S \in \mathbb{R}^{n \times s}$ with $s = O(k \log(k/\delta))$ such that, letting $\lambda = \frac{1}{k}\|A^{1/2} - A_k^{1/2}\|_F^2$:

$$\frac{1}{2}(A + \lambda I) \preceq \left(A^{1/2} S S^T A^{1/2} + \lambda I\right) \preceq \frac{3}{2}(A + \lambda I)$$

where $M \preceq N$ indicates $x^T M x \leq x^T N x$ for all x. If we set $\tilde{\tau}_i^k(A^{1/2}) = 2 \cdot x_i^T \left(A^{1/2} S S^T A^{1/2} + \lambda I\right)^+ x_i$, where x_i is the i^{th} column of $A^{1/2}$ we have the desired bound. Of course, we cannot directly compute this value without factoring A to form $A^{1/2}$. However, as shown in Lemma 6 of [42]:

$$\begin{aligned} &x_i^T (A^{1/2} S S^T A^{1/2} + \lambda I)^{-1} x_i \\ &\qquad = \frac{1}{\lambda}\left(A - AS(S^T A S + \lambda I)^{-1} S^T A\right)_{i,i}. \end{aligned}$$

Computing $(S^T A S + \lambda I)^{-1}$ requires accessing A $O(s^2) = O((k \log(k/\delta))^2)$ times and runtime $O(s^\omega) = O((k \log(k/\delta))^\omega)$. Computing all n diagonal entries of $AS(S^T A S + \lambda I)^{-1} S^T A$ then requires $O(nk \log(k/\delta))$ accesses to A and $O(n(k \log(k/\delta))^{\omega - 1})$ time. With these entries in hand we can simply subtract from the diagonal entries of A and rescale to give the final leverage score approximation. Critically, this calculation always reads *all diagonal entries* of A, allowing it to identify rows containing large off diagonal entries and skirt the $\text{nnz}(A)$ time lower bound for general matrices.

Note that the stated runtime in [42] for outputting S is $\tilde{O}(nk)$ accesses to A (*kernel evaluations* in the language of [42]) and $\tilde{O}(nk^2)$ runtime. However this runtime is improved to $\tilde{O}(nk^{\omega - 1})$ using fast matrix multiplication. ∎

In order to apply Lemmas 3 and 4 to low-rank approximation of A, we now show that the ridge leverage scores of $A^{1/2}$ coarsely approximate those of A:

Lemma 5 (Ridge Leverage Score Bound). *For any PSD matrix $A \in \mathbb{R}^{n \times n}$:*

$$\tau_i^k(A) \leq 2\sqrt{\frac{n}{k}} \cdot \tau_i^k(A^{1/2}).$$

Proof: We write $A^{1/2}$ in its eigendecomposition $A^{1/2} = U\Lambda^{1/2}U^T$, where $\Lambda_{i,i} = \lambda_i$ is the i^{th} eigenvalue of A.

Letting x_i denote the i^{th} column of $A^{1/2}$ we have:

$$\begin{aligned}\tau_i^k(A^{1/2}) &= x_i^T \left(A + \frac{\|A^{1/2} - A_k^{1/2}\|_F^2}{k} I \right)^{-1} x_i \\ &= x_i^T U \bar{\Lambda} U^T x_i\end{aligned}$$

where $\bar{\Lambda}_{i,i} \stackrel{\text{def}}{=} \frac{1}{\lambda_i + \frac{1}{k}\sum_{j=k+1}^n \lambda_j}$. We can similarly write:

$$\begin{aligned}\tau_i^k(A) &= a_i^T \left(A^2 + \frac{\|A - A_k\|_F^2}{k} I \right)^{-1} a_i \\ &= x_i^T A^{1/2} \left(A^2 + \frac{\|A - A_k\|_F^2}{k} I \right)^{-1} A^{1/2} x_i \\ &= x_i^T U \hat{\Lambda} U^T x_i\end{aligned}$$

where $\hat{\Lambda}_{i,i} \stackrel{\text{def}}{=} \frac{\lambda_i}{\lambda_i^2 + \frac{1}{k}\sum_{j=k+1}^n \lambda_j^2}$. Showing $\hat{\Lambda} \preceq 2\sqrt{\frac{n}{k}} \cdot \bar{\Lambda}$ is enough to give the lemma. Specifically we must show, for all i, $\hat{\Lambda}_{i,i} \leq 2\sqrt{\frac{n}{k}} \cdot \bar{\Lambda}_{i,i}$ which after cross-multiplying is equivalent to:

$$\lambda_i^2 + \frac{1}{k}\lambda_i \sum_{j=k+1}^n \lambda_j \leq 2\sqrt{\frac{n}{k}} \left(\lambda_i^2 + \frac{1}{k} \sum_{j=k+1}^n \lambda_j^2 \right). \quad (3)$$

First consider the relatively large eigenvalues. Say we have $\frac{1}{k}\sum_{j=k+1}^n \lambda_j \leq \sqrt{\frac{n}{k}}\lambda_i$. Then:

$$\lambda_i^2 + \frac{1}{k}\lambda_i \sum_{j=k+1}^n \lambda_j \leq \left(1 + \sqrt{n/k}\right) \lambda_i^2$$

which gives (3). Next consider small eigenvalues with $\frac{1}{k}\sum_{j=k+1}^n \lambda_j \geq \sqrt{\frac{n}{k}}\lambda_i$. In this case:

$$\begin{aligned}\lambda_i^2 + \frac{1}{k}\lambda_i \sum_{j=k+1}^n \lambda_j &\leq \lambda_i^2 + \frac{1}{\sqrt{n} \cdot k^{3/2}} \left(\sum_{j=k+1}^n \lambda_j \right)^2 \\ &\leq \lambda_i^2 + \frac{1}{\sqrt{n} \cdot k^{3/2}} \cdot n \sum_{j=k+1}^n \lambda_j^2 \\ &\qquad \text{(Norm bound: } \|\cdot\|_1^2 \leq n \|\cdot\|_2^2) \\ &\leq \sqrt{\frac{n}{k}} \left(\lambda_i^2 + \frac{1}{k} \sum_{j=k+1}^n \lambda_j^2 \right)\end{aligned}$$

which gives (3), completing the proof. ■

Combining Lemmas 2, 3, 4, 5 we have:

Corollary 6 (Fast PSD Ridge Leverage Score Sampling)**.** *There is an algorithm that given any PSD matrix $A \in \mathbb{R}^{n \times n}$ runs in $\tilde{O}(nk^{\omega-1})$ time, accesses $\tilde{O}(nk)$ entries of A, and with prob. $1-\delta$ outputs a weighted sampling matrix $S_1 \in \mathbb{R}^{n \times \tilde{O}\left(\frac{\sqrt{nk}}{\epsilon^2}\right)}$ such that AS_1 is an (ϵ,k)-column PCP of A.*

Proof: By Lemma 4 we can compute constant factor approximations to the ridge leverage scores of $A^{1/2}$ in time $\tilde{O}(nk^{\omega-1})$. Applying Lemma 5, if we scale these scores up by $2\sqrt{n/k}$ they will be overestimates of the ridge leverage scores of A. If we set $t = O\left(\frac{\log(k/\delta)}{\epsilon^2} \cdot \sum \tilde{\tau}_i^k\right)$, and generate S_1 by sampling t columns of A with probabilities proportional to these estimated scores, by Lemma 3, AS_1 will be an (ϵ,k)-column PCP of A with probability $1-\delta$. By Lemma 2, $\sum_{i=1}^n \tau_i^k(A^{1/2}) \leq 2k$. So we have $t = \tilde{O}(\sum \tilde{\tau}_i^k/\epsilon^2) = \tilde{O}(\sqrt{nk}/\epsilon^2)$. ■

Forming AS_1 requires reading just $\tilde{O}(n^{3/2}\sqrt{k}/\epsilon^2)$ entries of A. At this point, we could employ any input sparsity time algorithm to find a near optimal rank-k projection P for approximating AS_1 in $O(\text{nnz}(AS_1)) + n\,\text{poly}(k/\epsilon) = n^{3/2} \cdot \text{poly}(k/\epsilon)$ time. This would in turn yield a near optimal low-rank approximation of A. However, as we will see in the next section, by further sampling the rows of AS_1, we can significantly improve this runtime.

III. Row Sampling

To achieve near linear dependence on n, we sample roughly $\sqrt{nk}$ rows from AS_1, producing an even smaller matrix $S_2^T AS_1$, which we can fully read and from which we can form a near optimal low-rank approximation to AS_1 and consequently to A. However, sampling AS_1 is challenging: we cannot employ input sparsity time methods as we cannot afford to read the full matrix, and since it is no longer PSD, we cannot apply the same approach we used for A, approximating the ridge leverage scores with those of $A^{1/2}$.

Rewriting Definition 1 using the SVD $AS_1 = U\Sigma V^T$ (and transposing AS_1 to give row instead of column scores) we see that the row ridge leverage scores of AS_1 are the diagonal entries of:

$$AS_1 \left(S_1^T A^T A S_1 + \frac{\|AS_1 - (AS_1)_k\|_F^2}{k} I \right)^+ S_1^T A^T = U \bar{\Sigma} U^T$$

where $\bar{\Sigma}_{i,i} = \frac{\Sigma_{i,i}^2}{\Sigma_{i,i}^2 + \frac{\|AS_1 - (AS_1)_k\|_F^2}{k}}$. That is, the row ridge leverage scores depend only on the column span U of AS_1 and its spectrum. Since AS_1 is a column PCP of A this gives hope that the two matrices have similar leverage scores.

Unfortunately, this is *not the case*. It is possible to have rows in AS_1 with ridge leverage scores significantly higher than in A. Thus, even if we knew the ridge leverage scores of A, we would have to scale them up significantly to sample from AS_1. As an example, consider A with relatively uniform ridge leverage scores: $\tau_i(A) \approx k/n$ for all i. When a column is selected to be included in AS_1 it will be reweighted by roughly a factor of $\sqrt{n/k}$. Now, append a number of rows to A each with very small norm and just a containing single non-zero entry. These rows will have little effect on the ridge leverage scores if their norms are small enough. However, if the column corresponding to the nonzero in a row is selected, the row will appear in AS_1 with $\sqrt{n/k}$ times the weight that it appears in A, and its ridge leverage score will be roughly a factor n/k times higher.

Fortunately, we are still able to show that sampling the rows of AS_1 by the rank $k' = O(k/\epsilon)$ leverage scores of

$A^{1/2}$ scaled up by a $\sqrt{n/k'}$ factor yields a row PCP for this matrix. Our proof works not with the ridge scores of AS_1 but with the *standard leverage scores* of a near optimal low-rank approximation to this matrix – specifically the approximation given by projecting onto the top eigenvectors of A. We have:

Lemma 7 (Row PCP)**.** *For any PSD $A \in \mathbb{R}^{n\times n}$ and $\epsilon \leq 1$ let $k' = \lceil ck/\epsilon \rceil$ and let $\tilde{\tau}_i^{k'}(A^{1/2}) \geq \tau_i^{k'}(A^{1/2})$ be an overestimate for the i^{th} rank-k' ridge leverage score of $A^{1/2}$. Let $\tilde{\ell}_i = \sqrt{\frac{16n\epsilon}{k}} \cdot \tilde{\tau}_i^{k'}(A^{1/2})$, $p_i = \frac{\tilde{\ell}_i}{\sum_i \tilde{\ell}_i}$, and $t = \frac{c' \log n}{\epsilon^2} \sum_i \tilde{\ell}_i$. Construct weighted sampling matrices $S_1, S_2 \in \mathbb{R}^{n\times t}$ each whose j^{th} column is set to $\frac{1}{\sqrt{tp_i}} e_i$ with probability p_i. For sufficiently large constants c, c', with probability $\frac{99}{100}$, letting $\tilde{A} = S_2^T A S_1$, for any rank-k orthogonal projection $P \in \mathbb{R}^{t\times t}$:*

$$\begin{aligned}(1-\epsilon)\|AS_1(I-P)\|_F^2 &\leq \|\tilde{A}(I-P)\|_F^2 + \Delta \\ &\leq (1+\epsilon)\|AS_1(I-P)\|_F^2\end{aligned}$$

for some fixed Δ (independent of P) with $|\Delta| \leq 600\|A - A_k\|_F^2$. We refer to $\tilde{A}$ as an (ϵ, k)-row PCP of AS_1.

Note that by Lemma 2, $\sum_i \tau_i^{k'}(A^{1/2}) = O(k/\epsilon)$. So if $\tilde{\tau}_i^{k'}(A^{1/2})$ is a constant factor approximation to $\tau_i^{k'}(A^{1/2})$, $t = O\left(\frac{\sqrt{nk}\log n}{\epsilon^{2.5}}\right)$. Also note that the Lemma requires both S_1 and S_2 to be sampled using the rank k' ridge scores. If we sample S_1 using a sum of the rank-k and rank-k' ridge scores (appropriately scaled) Lemma 7 and Lemma 3 will hold simultaneously.

By applying an input sparsity time low-rank approximation algorithm to $\tilde{A}$ (which has just $\tilde{O}\left(\frac{nk}{\epsilon^5}\right)$ entries) we can find a near optimal low-rank approximation of AS_1, and thus for A. However, in our final algorithm, we will take a somewhat different approach. We are able to show that using appropriate sampling probabilities, we can in fact sample $\tilde{A}$ which is a projection-cost preserving sketch of AS_1 for *spectral norm* error. As we will see, recovering a near optimal spectral norm low-rank approximation to AS_1 suffices to find a near optimal Frobenius norm approximation to A, and lets us improve ϵ dependencies in our final runtime.

Lemma 8 (Spectral Norm Row PCP)**.** *For any PSD $A \in \mathbb{R}^{n\times n}$, and $\epsilon < 1$ let $k' = \lceil ck/\epsilon^2 \rceil$ and $\tilde{\tau}_i^{k'}(A^{1/2}) \geq \tau_i^{k'}(A^{1/2})$ be an overestimate for the i^{th} rank-k' ridge leverage score of $A^{1/2}$. Let $\tilde{\ell}_i = 4\epsilon\sqrt{\frac{n}{k}}\tau_i^{k'}(A^{1/2})$, $p_i = \frac{\tilde{\ell}_i}{\sum_i \tilde{\ell}_i}$, and $t = \frac{c' \log n}{\epsilon^2} \cdot \sum_i \tilde{\ell}_i$. Construct weighted sampling matrices $S_1, S_2 \in \mathbb{R}^{n\times t}$, each whose j^{th} column is set to $\frac{1}{\sqrt{tp_i}} e_i$ with probability p_i. For sufficiently large constants c, c', with high probability (i.e. probability $\geq 1 - 1/n^d$ for some large constant d), letting $\tilde{A} = S_2^T A S_1$, for any orthogonal projection $P \in \mathbb{R}^{t\times t}$:*

$$\begin{aligned}(1-\epsilon)\|AS_1(I-P)\|_2^2 - \frac{\epsilon}{k}\|A - A_k\|_F^2 &\leq \|\tilde{A}(I-P)\|_2^2 \\ &\leq (1+\epsilon)\|AS_1(I-P)\|_2^2 + \frac{\epsilon}{k}\|A - A_k\|_F^2.\end{aligned}$$

We refer to $\tilde{A}$ as an (ϵ, k)-spectral PCP of AS_1.

Note that if $\tilde{\tau}_i^{k'}(A^{1/2})$ is a constant factor approximation to $\tau_i^{k'}(A^{1/2})$, $t = O\left(\frac{\sqrt{nk}\log n}{\epsilon^3}\right)$. The proofs of Lemmas 7 and 8 are given in Appendix B of the our full paper.

IV. Full Low-Rank Approximation Algorithm

We are finally ready to give our main algorithm for relative error low-rank approximation of PSD matrices in $\tilde{O}(n \operatorname{poly}(k/\epsilon))$ time, Algorithm 1. We set $k_1 \stackrel{\text{def}}{=} \lceil ck/\epsilon \rceil$ and estimate the both the rank-k and rank-$c'k_1$ ridge leverage scores of $A^{1/2}$ using the algorithm of [42] (Step 1). If c, c' are sufficiently large, sampling by the sum of these scores (Steps 2-3) ensures that AS_1 is an (ϵ, k)-column PCP for A and simultaneously, by applying Lemmas 7 and 8 that $\tilde{A} = S_2^T A S_1$ is a row PCP in both spectral and Frobenius norm with rank k_1 and error $\epsilon = 1/2$ for AS_1

In conjunction, these guarantees ensure that we can apply an input sparsity time algorithm to $\tilde{A}$ (Step 4) to find a rank-k_1 Z satisfying $\|AS_1 - AZZ^T\|_2^2 = O(\|AS_1 - (AS_1)_{k_1}\|_2^2 + \frac{1}{k_1}\|A - A_k\|_F^2) = O\left(\frac{\epsilon}{k}\|A - A_k\|_F^2\right)$, where the final bound holds since $k_1 = \Theta(k/\epsilon)$. Due to this strong spectral norm bound, projecting AS_1 to Z and taking the best rank-k approximation in the span gives a near optimal Frobenius norm low-rank approximation to AS_1 and hence A.

We can still not afford to read AS_1 in its entirety, so we employ a number of standard leverage score sampling techniques to perform this projection approximately. In Step 5, we sample $\tilde{O}(k/\epsilon^2)$ columns of AS_1 using the leverage scores of Z (its row norms since it is an orthonormal matrix) to form AS_1S_3. We argue that there is a good rank-k approximation to AS_1 lying in both the column span of AS_1S_3 and the row span of Z^T. In Step 6 we find a near optimal such approximation by further sampling $\tilde{O}(k/\epsilon^4)$ rows AS_1 by the leverage scores of AS_1S_3 (the row norms of V, an orthonormal basis for its span), and computing the best rank-k approximation to the sampled matrix falling in the column span of AS_1S_3 and the row span of Z^T.

Finally, in Step 7 we approximately project A to the span of this rank-k approximation by first sampling by its leverage scores (the row norms of Q) and projecting.

Algorithm 1: **PSD Low-Rank Approximation**

```
1) Let $k_1 = \lceil ck/\epsilon \rceil$. For all $i \in [1,..,n]$ compute $\tilde{\tau}_i^k(A^{1/2})$ and $\tilde{\tau}_i^{c'k_1}(A^{1/2})$ which are constant factor approximations to the ridge leverage scores $\tau_i^k(A^{1/2})$ and $\tau_i^{c'k_1}(A^{1/2})$ respectively.
2) Set $\ell_i^{(1)} = \sqrt{\frac{n}{k}}\tilde{\tau}_i^k(A^{1/2}) + \sqrt{\frac{n\epsilon^4}{k_1}}\tilde{\tau}_i^{c'k_1}(A^{1/2})$ and $\ell_i^{(2)} = \sqrt{\frac{n}{k_1}}\tilde{\tau}_i^{c'k_1}(A^{1/2})$. Set $p_i^{(1)} = \frac{\ell_i^{(1)}}{\sum_i \ell_i^{(1)}}$ and $p_i^{(2)} = \frac{\ell_i^{(2)}}{\sum_i \ell_i^{(2)}}$.
3) Set $t_1 = \frac{c_1 \log n}{\epsilon^2} \sum_i \ell_i^{(1)}$ and $t_2 = c_2 \log n \sum_i \ell_i^{(2)}$. Sample $S_1 \in \mathbb{R}^{n\times t_1}$ whose $j^{th}$ column is set to $\frac{1}{\sqrt{tp_i^{(1)}}} e_i$ with probability $p_i^{(1)}$. Sample $S_2 \in \mathbb{R}^{n\times t_2}$ analogously with $p_i^{(2)}$.
```

4) Let $\tilde{A} = S_2^T A S_1$, and use an input sparsity time algorithm to compute orthonormal $Z \in \mathbb{R}^{t_1 \times k_1}$ satisfying the spectral guarantee $\|\tilde{A} - \tilde{A} Z Z^T\|_2^2 \leq 2\|\tilde{A} - \tilde{A}_{k_1}\|_2^2 + \frac{2}{k_1}\|\tilde{A} - \tilde{A}_{k_1}\|_F^2$.

5) Let $t_3 = c_3\left(\frac{k \log(k/\epsilon)}{\epsilon} + \frac{k}{\epsilon^2}\right)$, set $p_i^{(3)} = \frac{\|z_i\|_2^2}{\|Z\|_F^2}$, and sample $S_3 \in \mathbb{R}^{t_1 \times t_3}$ where the j^{th} column is set to $\frac{1}{\sqrt{t_3 p_i^{(3)}}} e_i$ with probability $p_i^{(3)}$. Compute $V \in \mathbb{R}^{n \times t_3}$ which is an orthogonal basis for the column span of $A S_1 S_3$.

6) Let $p_i^{(4)} = \frac{\|v_i\|_2^2}{\|V\|_F^2}$ and $t_4 = c_4\left(\frac{t_3 \log t_3}{\epsilon^2}\right)$. Sample $S_4 \in \mathbb{R}^{n \times t_4}$ where the j^{th} column is set to $\frac{1}{\sqrt{t_4 p_i^{(4)}}} e_i$ with probability $p_i^{(4)}$. Compute $W \in \mathbb{R}^{t_3 \times t_1}$ satisfying:

$$W = \underset{W \,|\, \operatorname{rank}(W) = k}{\arg\min} \|S_4^T A S_1 S_3 W Z^T - S_4^T A S_1\|_F^2.$$

7) Compute an orthogonal basis $Q \in \mathbb{R}^{n \times k}$ for the column span of $A S_1 S_3 W$. Let $t_5 = c_5\left(k \log k + \frac{k}{\epsilon}\right)$, set $p_i^{(5)} = \frac{\|q_i\|_2^2}{\|Q\|_F^2}$, where q_i is the i^{th} row of Q. Sample $S_5 \in \mathbb{R}^{n \times t_5}$ where the j^{th} column is set of $\frac{1}{\sqrt{t_5 p_i^{(5)}}} e_i$ with probability $p_i^{(5)}$. Solve:

$$N = \underset{N \in \mathbb{R}^{n \times k}}{\arg\min} \|S_5^T Q N^T - S_5^T A\|_F^2.$$

8) Return $Q, N \in \mathbb{R}^{n \times k}$.

A proof of correctness for Algorithm 1 is contained in our full paper, yielding:

Theorem 9 (Sublinear Time Low-Rank Approximation)**.** *Given any PSD* $A \in \mathbb{R}^{n \times n}$*, for sufficiently large constants* $c, c', c_1, c_2, c_3, c_4, c_5$*, for any* $\epsilon < 1$*, Algorithm 1 accesses* $O(\frac{n \cdot k \log^2 n}{\epsilon^{2.5}} + \sqrt{n} k^{1.5} \cdot \log^2 n \cdot \operatorname{poly}(1/\epsilon))$ *entries of* A*, runs in* $\tilde{O}\left(\frac{nk^{\omega-1}}{\epsilon^{2(\omega-1)}} + \sqrt{n} k^{\omega-.5} \cdot \operatorname{poly}(1/\epsilon)\right)$ *time, and with probability at least* $9/10$ *outputs* $M, N \in \mathbb{R}^{n \times k}$ *with:*

$$\|A - M N^T\|_F^2 \leq (1+\epsilon)\|A - A_k\|_F^2.$$

In many applications it is desirable that the low-rank approximation to A is also symmetric and positive semidefinite. We show in Appendix C of the full paper that a modification to Algorithm 1 can satisfy this constraint also in $\tilde{O}(n \operatorname{poly}(k/\epsilon))$ time. The upshot is:

Theorem 10 (Sublinear Time Low-Rank Approximation – PSD Output)**.** *There is an algorithm that given any PSD* $A \in \mathbb{R}^{n \times n}$*, accesses* $\tilde{O}\left(\frac{nk^2}{\epsilon^2} + \frac{nk}{\epsilon^3}\right)$ *entries of* A*, runs in* $\tilde{O}\left(\frac{nk^\omega}{\epsilon^\omega} + \frac{nk^{\omega-1}}{\epsilon^{3(\omega-1)}}\right)$ *time and with probability at least* $9/10$ *outputs* $M \in \mathbb{R}^{n \times k}$ *with:*

$$\|A - M M^T\|_F^2 \leq (1+\epsilon)\|A - A_k\|_F^2.$$

V. Query Lower Bound

We now present our lower bound on the number of accesses to A required to compute a near optimal low-rank approximation, matching the query complexity of Algorithm IV up to a $\tilde{O}(1/\epsilon^{1.5})$ factor.

Theorem 11. *Assume that* k, ϵ *are such that* $nk/\epsilon = o(n^2)$*. Any algorithm that given PSD* $A \in \mathbb{R}^{n \times n}$ *outputs a* $(1+\epsilon)$*-approximate rank-*k *approximation to* A *(in the Frobenius norm) with probability at least* $2/3$ *must read at least* $\Omega(nk/\epsilon)$ *positions of* A *in expectation.*

The idea behind Theorem 11 is to draw A from a distribution over binary matrices. A has all 1's on its diagonal, along with k randomly positioned (non-consecutive) blocks of all 1's, each of size $\sqrt{2\epsilon n/k} \times \sqrt{2\epsilon n/k}$. In other words, A is the adjacency matrix (plus identity) of a graph with k cliques of size $\sqrt{2\epsilon n/k}$, placed on random subsets of the vertices, with all other vertices isolated.

It is easy to see that A is PSD since applying a permutation yields a block diagonal matrix, each of whose blocks is a PSD matrix (either a single 1 entry or a rank-1 all 1's block). The optimal rank-k approximation to A projects off each of the k blocks, achieving Frobenius norm error $\|A - A_k\|_F^2 = n - k\sqrt{2\epsilon n/k} \approx n$. In order to match this up to a $1+\epsilon$ factor, any near optimal rank-k approximation must at least capture a constant fraction of the Frobenius norm mass in the blocks since this mass is $k \cdot 2\epsilon n/k = 2\epsilon n$.

Doing so requires identifying at least a constant fraction of the blocks. However, since block positions are chosen uniformly at random, and since the diagonal entries of A are identical and so convey no information about the positions, to identify a single block, any algorithm essentially must read arbitrary off diagonal entries until it finds a 1. There are $\approx n^2$ off diagonal entries with just $2\epsilon n$ of them 1's, so identifying a first block requires $\Omega(n/\epsilon)$ queries to A. Since the vast majority of vertices are isolated and not contained within a block, finding this first block does little to make finding future blocks easier. So overall, the algorithm must make $\Omega(nk/\epsilon)$ queries to find a constant fraction of the k blocks and output a near optimal low-rank approximation.

While the above intuition is the key idea behind the lower bound, a rigorous proof requires a number of additional steps and modifications, detailed in the remainder of this section.

A. Primitive Lower bound

We first prove a lower bound of $\Omega(n/\epsilon)$ accesses to A for a restricted class of algorithms, and then strengthen this to $\Omega(nk/\epsilon)$ for any k for every algorithm, via a reduction. Our stronger lower bound for every algorithm will use our lower bound for the restricted class of algorithms as a black box. We call our lower bound for restricted algorithms our *primitive lower bound.*

For our $\Omega(n/\epsilon)$ primitive lower bound, we assume that $n/\epsilon = o(n^2)$, as otherwise the bound becomes $\Omega(n^2)$ which is best possible. For our strengthened $\Omega(nk/\epsilon)$ lower bound for every algorithm, we assume that $nk/\epsilon = o(n^2)$, as otherwise again the bound becomes $\Omega(n^2)$.

Consider a distribution μ on $n \times n$ binary PSD matrices A. We choose a uniformly random subset S of $[n]$ where

$|S| = \sqrt{64\epsilon n}$, where we assume, w.l.o.g., that $|S|$ is an integer. We generate A by setting for each $i \neq j \in S$, $A_{i,j} = 1$. We then set $A_{i,i} = 1$ for all i and set all remaining entries of A to equal 0. It is clear that A is PSD – after a permutation is composed of an $|S| \times |S|$ all ones block and an $(n-|S|) \times (n-|S|)$ identity. Let ν be the distribution on $n \times n$ PSD matrices A which only has support on the identity matrix I. Let $\gamma = \mu/2 + \nu/2$. Note that we associate a random subset S with the sampling of a matrix A according to γ (as well as to μ); in case A is drawn from ν this set S is not used in the construction of A.

Definition 12. *A matrix A' is said to be ϵ-primitive for an A in the support of γ if the squared Frobenius norm of $A - A'$, restricted to entries in $([n] \setminus S)^2$, is at least $n - |S| - 8$. Furthermore, $\|A - A'\|_F^2 \leq (1+\epsilon)n$. Note that A' is allowed to have any rank.*

Lemma 13. *If A' is ϵ-primitive for an A in the support of μ, then A' is not ϵ-primitive for I.*

Proof: Using $|S| \geq 2$, which follows from assuming that $n/\epsilon = o(n^2)$, each matrix A in the support of μ has $|S|^2 - |S| \geq \frac{|S|^2}{2} = 32\epsilon n$ off-diagonal entries which are 1 on rows and columns indexed by S. The submatrix of A indexed by rows and columns in $[n] \setminus S$ is the identity I_t, where $t = n - |S| = n - \sqrt{64\epsilon n}$. By definition, any ϵ-primitive matrix A' for A has squared Frobenius norm error at least $t - 8$ on these coordinates. It follows that A' must have value at least $1/2$ on at least $16\epsilon n$ of the off-diagonal entries on rows and columns indexed by S. Otherwise, since there are at least $32\epsilon n$ off-diagonal entries which are 1 on rows and columns indexed by S, we would have

$$\|A - A'\|_F^2 > n - \sqrt{64\epsilon n} - 8 + \frac{1}{4} \cdot 16\epsilon n > n + \epsilon n,$$

contradicting that A' is ϵ-primitive for A. Here we used that $3\epsilon n \geq 8 + \sqrt{64\epsilon n}$. Note that since $n/\epsilon = o(n^2)$, we have $\epsilon n = \omega(1)$, so this inequality is valid.

If also A' were ϵ-primitive for I, then $\|A' - I\|_F^2 \leq (1+\epsilon)n$. The previous paragraph implies on the rows and columns indexed by S, the squared Frobenius norm of the difference between A' and I is at least $4\epsilon n$. On the remaining coordinates, by definition, the squared Frobenius norm of the difference between A' and I must be at least $n - \sqrt{64\epsilon n} - 8 > n - 3\epsilon n$. Hence $\|A' - I\|_F^2 > (1+\epsilon)n$, a contradiction. ■

Recall for distributions α and β supported on elements s of a finite set S, that the total variation distance $D_{TV}(\alpha, \beta) = \sum_{s \in S} |\alpha(s) - \beta(s)|$, where $\alpha(s)$ is the probability of s in distribution α.

Corollary 14. *Suppose there is an algorithm which, with probability at least $2/3$, over its random coin tosses and random A drawn from γ, outputs an ϵ-primitive matrix for A. Further, suppose the algorithm reads at most r positions of A, possibly adaptively. Let S be a random variable indicating the list of positions read and their corresponding values. Let $L(\mu)$ denote the distribution of S when $A \sim \mu$, and let $L(\nu)$ denote the distribution of S when $A \sim \nu$. Then*

$$D_{TV}(L(\mu), L(\nu)) \geq 1/3.$$

Proof: By Lemma 13, if the algorithm succeeds then its output can be used to decide if $A \sim \mu$ or if $A \sim \nu$. The success probability of any such algorithm is well-known (see, e.g., Proposition 2.58 of [50]) to be at most $1/2 + D_{TV}(L(\mu), L(\nu))/2$. Making this quantity at least $2/3$ and solving for $D_{TV}(L(\mu), L(\nu))$ proves the corollary. ■

We now state the main theorem of this subsection, which we strengthen in the next subsection.

Theorem 15. *Suppose there is an algorithm which, with probability at least $2/3$, over its random coin tosses and random A drawn from γ, outputs an ϵ-primitive matrix for A. Further, suppose the algorithm reads at most r positions of A, possibly adaptively. Then $r = \Omega(n/\epsilon)$.*

Proof: By Corollary 14, and using the notation in that corollary, it suffices to show for $r = o(n/\epsilon)$, that $D_{TV}(L(\mu), L(\nu)) < 1/3$.

By Yao's minimax principle ([51], Theorem 3), we can assume the algorithm is deterministic, since if there is a randomized algorithm with the guarantees of the theorem, then by averaging there is also a deterministic one for some fixing of its random coin tosses. We can also assume the algorithm does not read the diagonal entries of A, since they are equal to 1 in all matrices in the support of both μ and ν. It follows that given that the algorithm reads a prefix of r zeros, then it always reads the same sequence $(i_1, j_1), \ldots, (i_r, j_r)$ of entries of A. Note that for $A \sim \nu$, all off-diagonal entries are 0 and so $L(\nu)((i_1, j_1, 0), \ldots, (i_r, j_r, 0)) = 1$. To show the claim then, it suffices to show that $L(\mu)((i_1, j_1, 0), \ldots, (i_r, j_r, 0)) \geq 2/3$, as then $D_{TV}(L(\mu), L(\nu)) < 1/3$. This is equivalent to showing that with probability at least $2/3$, over $A \sim \mu$, that $A_{i_1,j_1} = A_{i_2,j_2} = \cdots = A_{i_r,j_r} = 0$. For any $\ell \in [r]$,

$$\Pr[A_{i_\ell,j_\ell} = 0] = \frac{(|S|^2 - |S|)}{n^2 - n} \leq \frac{128\epsilon}{n},$$

assuming $n \geq 2$. By a union bound, for $r = o(n/\epsilon)$, we thus have with probability $1 - o(1)$, for $A \sim \mu$, $A_{i_1,j_1} = A_{i_2,j_2} = \cdots = A_{i_r,j_r} = 0$. This completes the proof. ■

B. Lower Bound for General Algorithms

In this subsection we remove the requirement that the low-rank approximation needs to be primitive, and simultaneously improve our lower bound to $\Omega(nk/\epsilon)$ whenever $nk/\epsilon = o(n^2)$. We will use our lower bound for primitive low-rank approximation as a black box.

W.l.o.g. assume n is divisible by k and partition n into k blocks. On the i-th block, $i = 1, \ldots, k$, we independently draw a random $n/k \times n/k$ PSD matrix A^i from γ, where n is replaced by n/k. Let S^i be a random subset of $\sqrt{64\epsilon n/k}$ indices in that block. Let γ_b be the distribution over $n \times n$ block matrices whose i^{th} block is A^i drawn as described.

Let $\epsilon' = \epsilon/10$. Consider an algorithm which outputs a $(1+\epsilon')$ rank-k approximation B to $A \sim \gamma_b$, with probability at least $2/3$ over the random choice of A and the random coins of the algorithm. Again by Yao's minimax principle we can assume that the algorithm is deterministic. Towards a contradiction, we also assume that the algorithm always reads $o(kn/\epsilon') = o(kn/\epsilon)$ entries of any input matrix. By construction, for any $A \sim \gamma_b$ there is a rank-k approximation of cost at most n; indeed this follows by choosing the best rank-1 solution for each block. Consequently if the algorithm succeeds, then the output B satisfies $\|B - A\|_F^2 \leq n + \epsilon' n$.

Let B^i be the restriction of B to coordinates in the i-th block. The restriction of A to coordinates outside of $\cup_i S^i$ is an identity matrix and so the squared Frobenius norm cost of any rank-k approximation restricted to these coordinates is at least $n - k$. Let c_i be the squared Frobenius norm cost of B^i for A^i restricted to the coordinates in S^i. Note that c_i is a random variable. Then, $n - k + \sum_{i=1}^k c_i \leq \sum_{i=1}^k \|B^i - A^i\|_F^2 \leq \|B - A\|_F^2 \leq n + \epsilon' n$, and so by averaging for at least a $7/8$ fraction of the blocks i, $c_i \leq 8 + 8\epsilon' n/k \leq 9\epsilon' n/k$, assuming $\epsilon' n/k \geq 8$, which holds if $\epsilon n/k = \omega(1)$, which follows from our assumption that $nk/\epsilon = o(n^2)$.

Let b_i be the squared Frobenius norm cost of B^i for A^i restricted to coordinates outside S^i. Here b_i is a random variable and $\sum_i b_i \geq n - k$. By averaging, for at least a $7/8$ fraction of the i, $b_i \geq n/k - 8$.

It follows by a union bound that for a uniformly random chosen block i, with probability at least $3/4$ over $A \sim \gamma_b$, we have $c_i \leq 9\epsilon' n/k$ and $b_i \geq n/k - 8$. We also have for a uniformly random block i, that with probability $1 - o(1)$ over $A \sim \gamma_b$, the number of entries read in the block is $o(n/\epsilon') = o(n/\epsilon)$. The latter follows by a Markov bound given that the total number of entries is $o(nk/\epsilon')$ by assumption. By a union bound, there exists a block i^* for which with probability at least $3/4 - o(1) > 2/3$ over $A \sim \gamma_b$, we have (1) $c_{i^*} \leq 9\epsilon' n/k$ and $b_{i^*} \geq n/k - 8$, and (2) the algorithm reads $o(n/\epsilon)$ entries of the block.

It follows that with probability at least $2/3$, the output B^{i^*} satisfies $\|B^{i^*} - A^{i^*}\|_F^2 \leq 9\epsilon' n/k + n/k \leq (1+\epsilon)n/k$, and $b_{i^*} \geq n/k - 8$, and so B^{i^*} is ϵ-primitive for A^{i^*}.

To obtain a contradiction to Theorem 15, we perform the following simulation. Given an $n/k \times n/k$ PSD matrix $\tilde{A} \sim \gamma$, we create an $n \times n$ PSD block matrix A by setting $A^{i^*} = \tilde{A}$ and by independently sampling A^j for all $j \neq i^*$ according to γ. We then run the above algorithm and output B^{i^*}. For each entry of A^j the above algorithm reads for some $j \neq i^*$, we can input that entry to the algorithm without reading any entry of $\tilde{A}$. If the algorithm reads an entry of A^{i^*} then we read the corresponding entry of $\tilde{A}$. If the algorithm ever reads more than $o(n/\epsilon)$ entries of $\tilde{A}$, then we abort. By the above, with probability at least $2/3$, B^{i*} is ϵ-primitive for $\tilde{A}$. Finally, to contradict Theorem 15 we need to ensure that $(n/k)/\epsilon = o((n/k)^2)$, which is implied by our assumption that $nk/\epsilon = o(n^2)$.

Thus we have contradicted Theorem 15, and our main lower bound theorem follows by rescaling ϵ by a constant:

Theorem 16. *Let $nk/\epsilon = o(n^2)$. Suppose there is an algorithm which, with probability at least $2/3$, over its random coin tosses and random A drawn from γ_b, outputs a $(1+\epsilon)$-approximate rank-k approximation to A. Further, suppose the algorithm reads at most r positions of A, possibly adaptively. Then $r = \Omega(nk/\epsilon)$.*

Theorem 11 follows immediately from the above.

VI. Spectral Norm Error Bounds

We conclude by discussing how to modify Algorithm 1 to output low-rank B achieving the spectral norm guarantee:

$$\|A - B\|_2^2 \leq (1+\epsilon)\|A - A_k\|_2^2 + \frac{\epsilon}{k}\|A - A_k\|_F^2. \qquad (4)$$

This can be significantly stronger than the Frobenius guarantee (1) when $\|A - A_k\|_F^2$ is large, and e.g., is critical in our application to sublinear time ridge regression.

Since additive error in the Frobenius norm upper bounds additive error in the spectral norm (see e.g. Theorem 3.4 of [52]), for B satisfying the Frobenius norm guarantee $\|A - B\|_F^2 \leq (1+\epsilon)\|A - A_k\|_F^2$, we immediately have the spectral bound $\|A - B\|_2^2 \leq \|A - A_k\|_2^2 + \epsilon\|A - A_k\|_F^2$. Thus, we can achieve 4 simply by running Algorithm 1 with error parameter ϵ/k. However, this approach is suboptimal. Applying Theorem 9, our query complexity would be $\Theta\left(\frac{nk^{3.5}\log^2 n}{\epsilon^{2.5}}\right)$. We improve this k dependence significantly in Algorithm 2. Since (4) is often applied with $k' = k/\epsilon$ and $\epsilon = \Theta(1)$ to give $\|A - B\|_2^2 \leq O\left(\frac{\epsilon}{k}\|A - A_k\|_F^2\right)$, optimizing k dependence is especially important.

We first give an extension of Lemma 3 to the spectral norm case. This lemma provides the column sampling analog to Lemma 8. It is proven in our full paper.

Lemma 17 (Spectral Norm PCP)**.** *For any $A \in \mathbb{R}^{n \times d}$, for $i \in \{1, \ldots, d\}$, let $\tilde{\tau}_i^k \geq \tau_i^k(A)$ be an overestimate for the i^{th} rank-k ridge leverage score. Let $p_i = \frac{\tilde{\tau}_i^k}{\sum_i \tilde{\tau}_i^k}$ and $t = \frac{c\log(k/\delta)}{\epsilon^2}\sum_i \tilde{\tau}_i^k$ for any $\epsilon < 1$ and sufficiently large constant c. Construct C by sampling t columns of A, each set to $\frac{1}{\sqrt{tp_i}}a_i$ with probability p_i. With probability $1 - \delta$, for any orthogonal projection $P \in \mathbb{R}^{n \times n}$: $(1-\epsilon)\|A - PA\|_2^2 - \frac{\epsilon}{k}\|A - A_k\|_F^2 \leq \|C - PC\|_2^2 \leq (1+\epsilon)\|A - PA\|_2^2 + \frac{\epsilon}{k}\|A - A_k\|_F^2$. We refer to C as an (ϵ, k)-spectral PCP of A.*

A. Spectral Norm Low-Rank Approximation Algorithm

We now apply Lemmas 8 and 17 to give our spectral norm low-rank approximation algorithm, Algorithm 2.

In Steps 1-3 we sample both rows and columns of A via the rank $\Theta(k/\epsilon)$ ridge leverage scores of $A^{1/2}$, ensuring with high probability that AS_1 is an (ϵ, k)-spectral PCP of A and $\tilde{A}$ is in turn an (ϵ, k)-spectral row PCP of AS_1. Thus, if we compute (in input sparsity time) a span Z which gives a near optimal spectral norm low-rank approximation to $\tilde{A}$ (Step 3), this span will also be nearly optimal for AS_1.

We approximately project AS_1 to Z by further sampling its columns using Z's leverage scores (Step 4). We use leverage score sampling again in Step 5 to approximately project A to the span of the result. This yields our final approximation, using that AS_1 is a spectral PCP for A.

Algorithm 2: **Low-Rank Approximation, Spectral**

```
1) Let $k_1 = \lceil ck/\epsilon^2 \rceil$. For all $i \in [1,..,n]$ compute $\tilde{\tau}_i^{k_1}(A^{1/2})$
   which is a constant factor approximation to $\tau_i^{k_1}(A^{1/2})$.
2) Set $\ell_i^{(1)} = 4\epsilon\sqrt{\frac{n}{k}}\tilde{\tau}_i^{k_1}(A^{1/2})$. Set $p_i^{(1)} = \frac{\ell_i^{(1)}}{\sum_i \ell_i^{(1)}}$ and $t_1 =
   \frac{c_1 \log n}{\epsilon^2}\sum_i \ell_i^{(1)}$. Sample $S_1, S_2 \in \mathbb{R}^{n \times t_1}$ each whose $j^{th}$
   column is set to $\frac{1}{\sqrt{tp_i^{(1)}}}e_i$ with probability $p_i^{(1)}$.
3) Let $\tilde{A} = S_2^T AS_1$, and use an input sparsity time algorithm
   to compute orthonormal $Z \in \mathbb{R}^{t_1 \times k}$ satisfying both $\|\tilde{A} -
   \tilde{A}ZZ^T\|_F^2 \leq 2\|\tilde{A} - \tilde{A}_k\|_F^2$ along with $\|\tilde{A} - \tilde{A}ZZ^T\|_2^2 \leq
   (1+\epsilon)\|\tilde{A} - \tilde{A}_k\|_2^2 + \frac{\epsilon}{k}\|\tilde{A} - \tilde{A}_k\|_F^2$.
4) Let $t_3 = c_3\left(k \log k + \frac{k^2}{\epsilon}\right)$, set $p_i^{(3)} = \frac{\|z_i\|_2^2}{\|Z\|_F^2}$, and sample
   $S_3 \in \mathbb{R}^{t_1 \times t_3}$ whose $j^{th}$ column is set to $\frac{e_i}{\sqrt{t_3 p_i^{(3)}}}$ with prob-
   ability $p_i^{(3)}$. Solve $M = \arg\min \|AS_1S_3 - MZ^TS_3\|_F^2$.
5) Compute a basis $Q \in \mathbb{R}^{n \times k}$ for the column span of $M$.
   Let $t_4 = c_4\left(k \log k + \frac{k^2}{\epsilon}\right)$, set $p_i^{(4)} = \frac{\|q_i\|_2^2}{\|Q\|_F^2}$ and sample
   $S_4 \in \mathbb{R}^{n \times t_4}$ where the $j^{th}$ column is set to $\frac{1}{\sqrt{t_4 p_i^{(4)}}}e_i$ with
   probability $p_i^{(4)}$. Solve $N = \arg\min \|S_4^T QN^T - S_4^T A\|_F^2$.
6) Return $Q, N \in \mathbb{R}^{n \times k}$.
```

In our full paper we formally analyze Algorithm 2, giving:

Theorem 18 (Sublinear Time Low-Rank Approximation –Spectral Norm Error)**.** *Given any PSD* $A \in \mathbb{R}^{n \times n}$*, for sufficiently large constants* c, c_1, c_2, c_3, c_4*, Algorithm 2 accesses* $O(\frac{n \cdot k \log^2 n}{\epsilon^6} + \frac{nk^2}{\epsilon})$ *entries of* A*, runs in* $\tilde{O}\left(\frac{nk^\omega}{\epsilon} + \frac{nk}{\epsilon^6} + (\sqrt{n}k^{\omega-1} + k^{\omega+1}) \cdot \text{poly}(1/\epsilon)\right)$ *time and with probability at least* $9/10$ *outputs* $M, N \in \mathbb{R}^{n \times k}$ *with* $\|A - MN^T\|_2^2 \leq (1+\epsilon)\|A - A_k\|_2^2 + \frac{\epsilon}{k}\|A - A_k\|_F^2$.

Theorem 18 can be leveraged to give a sublinear time, relative error algorithm for approximately solving the ridge regression problem $\min_{x \in \mathbb{R}^n} \|Ax - y\|_2^2 + \lambda\|x\|_2^2$ for PSD A. In our full paper we show that for sufficiently large k, any $O(1)$ optimal rank-k approximation to A can be used to solve ridge regression to relative error. This approximation can be computed in sublinear time via Algorithm 2 yielding:

Theorem 19 (Sublinear Time Ridge Regression)**.** *Given any PSD* $A \in \mathbb{R}^{n \times n}$*,* $\lambda \geq 0$*,* $y \in \mathbb{R}^n$*, and upper bound* $\tilde{s}_\lambda$ *on the statistical dimension* $s_\lambda \overset{\text{def}}{=} \text{tr}((A^2 + \lambda I)^{-1}A^2)$*, there is an algorithm accessing* $\tilde{O}\left(\frac{n\tilde{s}_\lambda^2}{\epsilon^4}\right)$ *entries of* A *and running in* $\tilde{O}\left(\frac{n\tilde{s}_\lambda^\omega}{\epsilon^{2\omega}}\right)$ *time, which outputs* $\tilde{x}$ *satisfying* $\|A\tilde{x} - y\|_2^2 + \lambda\|\tilde{x}\|_2^2 \leq (1+\epsilon) \cdot \min_{x \in \mathbb{R}^n} \|Ax - y\|_2^2 + \lambda\|x\|_2^2$.

When $\tilde{s}_\lambda \ll n$ as is often the case, the above significantly improves upon state-of-the-art input sparsity time runtimes for general matrices [44]. For a proof see our full paper.

Acknowledgments: The authors thank IBM Almaden where part of this work was done. D. Woodruff also thanks the Simons Institute program on Machine Learning and the XDATA program of DARPA for support.

References

[1] P. Drineas, A. M. Frieze, R. Kannan, S. Vempala, and V. Vinay, “Clustering large graphs via the singular value decomposition,” *Machine Learning*, vol. 56, no. 1-3, 2004.

[2] D. Feldman, M. Schmidt, and C. Sohler, “Turning big data into tiny data: Constant-size coresets for k-means, PCA, and projective clustering,” in *Proc. of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2013.

[3] Y. Liang, M. Balcan, V. Kanchanapally, and D. P. Woodruff, “Improved distributed principal component analysis,” in *Advances in Neural Information Processing Systems 27 (NIPS)*, 2014, pp. 3113–3121.

[4] M. B. Cohen, S. Elder, C. Musco, C. Musco, and M. Persu, “Dimensionality reduction for k-means clustering and low rank approximation,” in *Proc. of the 47th Annual ACM Symposium on Theory of Computing (STOC)*, 2015.

[5] Y. Azar, A. Fiat, A. R. Karlin, F. McSherry, and J. Saia, “Spectral analysis of data,” in *Proc. of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, 2001.

[6] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala, “Latent semantic indexing: A probabilistic analysis,” *Journal of Computer and System Sciences*, 2000.

[7] D. Achlioptas and F. McSherry, “On spectral learning of mixtures of distributions,” in *Proc. of the 18th Annual Conference on Computational Learning Theory*, 2005.

[8] R. Kannan, H. Salmasian, and S. Vempala, “The spectral method for general mixture models,” *SIAM Journal on Computing*, vol. 38, no. 3, pp. 1141–1156, 2008.

[9] P. Drineas, I. Kerenidis, and P. Raghavan, “Competitive recommendation systems,” in *Proc. of the 34th Annual ACM Symposium on Theory of Computing (STOC)*, 2002.

[10] T. Hofmann, “Collaborative filtering via Gaussian probabilistic latent semantic analysis,” in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2003.

[11] D. Achlioptas, A. Fiat, A. R. Karlin, and F. McSherry, “Web search via hub synthesis,” in *Proc. of the 42nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2001, pp. 500–509.

[12] J. M. Kleinberg, “Authoritative sources in a hyperlinked environment,” *Journal of the ACM*, vol. 46, no. 5, 1999.

[13] A. M. Frieze, R. Kannan, and S. Vempala, “Fast Monte-Carlo algorithms for finding low-rank approximations,” *Journal of the ACM*, vol. 51, no. 6, pp. 1025–1041, 2004.

[14] D. Achlioptas and F. McSherry, “Fast computation of low-rank matrix approximations,” *Journal of the ACM*, 2007.

[15] Z. Song, D. P. Woodruff, and H. Zhang, “Sublinear time orthogonal tensor decomposition,” in *Advances in Neural Information Processing Systems 29 (NIPS)*, 2016.

[16] T. Sarlos, "Improved approximation algorithms for large matrices via random projections," in *Proc. of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2006, pp. 143–152.

[17] K. L. Clarkson and D. P. Woodruff, "Low rank approximation and regression in input sparsity time," in *Proc. of the 45th Annual ACM Symposium on Theory of Computing (STOC)*, 2013, pp. 81–90.

[18] J. Bourgain, S. Dirksen, and J. Nelson, "Toward a unified theory of sparse dimensionality reduction in Euclidean space," in *Proc. of the 47th Annual ACM Symposium on Theory of Computing (STOC)*, 2015, pp. 499–508.

[19] M. B. Cohen, "Nearly tight oblivious subspace embeddings by trace inequalities," in *Proc. of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2016.

[20] X. Meng and M. W. Mahoney, "Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression," in *Proc. of the 45th Annual ACM Symposium on Theory of Computing (STOC)*, 2013.

[21] J. Nelson and H. L. Nguyen, "OSNAP: Faster numerical linear algebra algorithms via sparser subspace embeddings," in *Proc. of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2013.

[22] D. P. Woodruff, "Sketching as a tool for numerical linear algebra," *Foundations and Trends in Theoretical Computer Science*, vol. 10, no. 1-2, pp. 1–157, 2014.

[23] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[24] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in *Proc. of the 45th Annual ACM Symposium on Theory of Computing (STOC)*, 2013, pp. 665–674.

[25] M. Hardt, "Understanding alternating minimization for matrix completion," in *Proc. of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2014.

[26] S. J. Young and E. R. Scheinerman, "Random dot product graph models for social networks," in *International Workshop on Algorithms and Models for the Web-Graph*, 2007.

[27] T. F. Cox and M. A. Cox, *Multidimensional scaling*, 2000.

[28] D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert, "Quantum state tomography via compressed sensing," *Physical review letters*, vol. 105, no. 15, p. 150401, 2010.

[29] A. M.-C. So and Y. Ye, "Theory of semidefinite programming for sensor network localization," in *Proc. of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2005, pp. 405–414.

[30] G. Young and A. S. Householder, "Discussion of a set of points in terms of their mutual distances," *Psychometrika*, vol. 3, no. 1, pp. 19–22, 1938.

[31] P. Drineas and M. W. Mahoney, "On the Nyström method for approximating a Gram matrix for improved kernel-based learning," *Journal of Machine Learning Research*, 2005.

[32] K. Zhang, I. W. Tsang, and J. T. Kwok, "Improved Nyström low-rank approximation and error analysis," in *Proc. of the 25th International Conference on Machine Learning*, 2008.

[33] S. Kumar, M. Mohri, and A. Talwalkar, "Sampling techniques for the Nyström method." in *Proc. of the 12th International Conference on Artificial Intelligence and Statistics*, 2009.

[34] M.-A. Belabbas and P. J. Wolfe, "Spectral methods in machine learning and new strategies for very large datasets," *Proceedings of the National Academy of Sciences*, vol. 106, no. 2, pp. 369–374, 2009.

[35] M. Li, J. T.-Y. Kwok, and B. Lu, "Making large-scale Nyström approximation possible," in *Proc. of the 27th International Conference on Machine Learning*, 2010.

[36] A. Gittens and M. W. Mahoney, "Revisiting the Nyström method for improved large-scale machine learning," in *Proc. of the 30th International Conference on Machine Learning*, 2013, Preliminary version at arXiv:1303.1849.

[37] S. Wang and Z. Zhang, "Improving CUR matrix decomposition and the Nyström approximation via adaptive sampling." *Journal of Machine Learning Research*, vol. 14, no. 1, 2013.

[38] X. Duan, J. Li, Q. Wang, and X. Zhang, "Low rank approximation of the symmetric positive semidefinite matrix," *Journal of Computational and Applied Mathematics*, 2014.

[39] S. Wang, L. Luo, and Z. Zhang, "SPSD matrix approximation vis column selection: theories, algorithms, and extensions," *Journal of Machine Learning Research*, vol. 17, no. 49, 2016.

[40] J. A. Tropp, A. Yurtsever, M. Udell, and V. Cevher, "Randomized single-view algorithms for low-rank matrix approximation," *arXiv:1609.00048*, 2016.

[41] C. Li, S. Jegelka, and S. Sra, "Fast DPP sampling for Nyström with application to kernel methods," in *Proc. of the 33rd International Conference on Machine Learning*, 2016.

[42] C. Musco and C. Musco, "Recursive sampling for the Nyström method," *arXiv:1605.07583*, 2016.

[43] K. Clarkson and D. P. Woodruff, "Low-rank PSD approximation in input-sparsity time," in *Proc. of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2017.

[44] H. Avron, K. L. Clarkson, and D. P. Woodruff, "Sharper bounds for regression and low-rank approximation with regularization," 2016.

[45] A. Deshpande, L. Rademacher, S. Vempala, and G. Wang, "Matrix approximation and projective clustering via volume sampling," in *Proc. of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2006.

[46] A. Deshpande and S. Vempala, "Adaptive sampling and fast low-rank matrix approximation," in *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*. Springer, 2006, pp. 292–303.

[47] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, "Subspace sampling and relative-error matrix approximation: Column-row-based methods," in *European Symposium on Algorithms*. Springer, 2006, pp. 304–314.

[48] N. Anari, S. O. Gharan, and A. Rezaei, "Monte Carlo Markov chain algorithms for sampling strongly Rayleigh distributions and determinantal point processes," in *Proc. of the 29th Annual Conference on Computational Learning Theory*, 2016.

[49] M. B. Cohen, C. Musco, and C. Musco, "Input sparsity time low-rank approximation via ridge leverage score sampling," in *Proc. of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2017.

[50] Z. Bar-Yossef, "The complexity of massive data set computations," Ph.D. dissertation, UC Berkeley, 2002.

[51] A. C.-C. Yao, "Probabilistic computations: Toward a unified measure of complexity," in *Proc. of the 18th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 1977, pp. 222–227.

[52] M. Gu, "Subspace iteration randomization and singular value problems," *arXiv:1408.2208*, 2014.