Leverage Score Sampling for Tensor Product Matrices in Input Sparsity Time

David P. Woodruff [*1] **Amir Zandieh** [*2]

Abstract

We propose an input sparsity time sampling algorithm that can spectrally approximate the Gram matrix corresponding to the q-fold column-wise tensor product of q matrices using a nearly optimal number of samples, improving upon all previously known methods by $\text{poly}(q)$ factors. Furthermore, for the important special case of the q-fold self-tensoring of a dataset, which is the feature matrix of the degree-q polynomial kernel, the leading term of our method's runtime is proportional to the size of the input dataset and has no dependence on q. Previous techniques either incur $\text{poly}(q)$ slowdowns in their runtime or remove the dependence on q at the expense of having sub-optimal target dimension, and depend quadratically on the number of data-points in their runtime. Our sampling technique relies on a collection of q partially correlated random projections which can be simultaneously applied to a dataset X in total time that only depends on the size of X, and at the same time their q-fold Kronecker product acts as a near-isometry for any fixed vector in the column span of $X^{\otimes q}$. We also show that our sampling methods generalize to other classes of kernels beyond polynomial, such as Gaussian and Neural Tangent kernels.

1. Introduction

In many learning problems such as regression or PCA, one is given a *feature* (or design) matrix $\Phi \in \mathbb{R}^{m\times n}$ and needs to compute the inverse or singular value decomposition (SVD) of the Gram matrix $\Phi^\top\Phi$. However, the feature matrices Φ, particularly the features that correspond to kernel functions, often have a massive (sometimes infinite) number of rows, which makes the storage and computations involving $\Phi^\top\Phi$ prohibitively expensive. This has motivated a long line of work on approximating the Gram matrix $\Phi^\top\Phi$ by a low-rank matrix (Williams & Seeger, 2001; Rahimi & Recht, 2009; Avron et al., 2014; El Alaoui & Mahoney, 2014; Cohen et al., 2015; Musco & Musco, 2017; Avron et al., 2017).

In this work, we focus on feature matrices whose columns are tensor products of a large number of arbitrary vectors, i.e., $\Phi = X^{(1)} \otimes X^{(2)} \otimes \dots X^{(q)}$ for datasets $X^{(1)}, \dots X^{(q)} \in \mathbb{R}^{d\times n}$ (for tensor product notations see Definitions 2.1 and 2.2). Note that the tensor product matrix Φ defined this way has d^q rows and n columns. This type of tensor product feature matrix Φ is of great importance in learning applications, particularly because the special case of $X^{(1)} = \dots = X^{(q)}$ corresponds to the feature matrix of the degree-q polynomial kernel, i.e., the Gram matrix $\Phi^\top\Phi$ is the degree-q polynomial kernel matrix. To tackle scalability challenges, much work has focused on *compressing* the large number of rows of such tensor product feature matrices through linear sketching or sampling techniques (Pham & Pagh, 2013; Avron et al., 2014; Ahle et al., 2020; Meister et al., 2019; Zandieh et al., 2021; Song et al., 2021).

The aim of our work is to devise efficient sampling methods for reducing the dimensionality (number of rows) of tensor product matrices while preserving the spectral structure of the Gram matrix. Formally, for any given $\epsilon, \lambda > 0$ and any $X^{(1)}, \dots X^{(d)} \in \mathbb{R}^{d\times n}$, if the feature matrix is defined as $\Phi := X^{(1)} \otimes \dots X^{(q)}$, we want to find a sampling matrix $\Pi \in \mathbb{R}^{s\times d^d}$, such that the sub-sampled Gram matrix $\Phi^\top\Pi^\top\Pi\Phi$ is an (ϵ, λ)-*spectral approximation* to $\Phi^\top\Phi$, i.e.,

$$\frac{\Phi^\top\Phi + \lambda I}{1+\epsilon} \preceq \Phi^\top\Pi^\top\Pi\Phi + \lambda I \preceq \frac{\Phi^\top\Phi + \lambda I}{1-\epsilon}. \qquad (1)$$

Sampling a small number of rows of any matrix Φ according to its *leverage scores* is known to yield a spectral approximation to $\Phi^\top\Phi$ (Li et al., 2013). Our goal is to generate a sampling matrix Π according to the ridge leverage scores of Φ in input sparsity time, i.e., $O\left(\sum_{j=1}^q \text{nnz}\left(X^{(j)}\right)\right)$.

1.1. Our Main Results

• It is well-known that for any linear sketch or sampling matrix Π to satisfy (1), its number s of rows needs to be proportional to the *statistical dimension* $s_\lambda :=$

[*]Equal contribution [1]Max-Planck-Institut für Informatik [2]Carnegie Mellon University. Correspondence to: David Woodruff <dwoodruf@cs.cmu.edu>, Amir Zandieh <azandieh@mpi-inf.mpg.de>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

$\sum_{i=1}^{n} \frac{\lambda_i}{\lambda_i+\lambda}$, where the λ_i are the eigenvalues of $\Phi^\top\Phi$, (Avron et al., 2019). Woodruff & Zandieh (2020) recently showed that it is possible to generate a sampling matrix with $s = O\left(\frac{s_\lambda}{\epsilon^2}\log n\right)$ rows that satisfies (1) in time $\widetilde{O}\left(\mathrm{poly}(q,\epsilon^{-1})\cdot s_\lambda^2 n + q^{1.5}\sum_{j=1}^{q}\mathrm{nnz}\left(X^{(j)}\right)\right)$. The significance of this result was showing the possibility of decoupling ϵ^{-1} factors from the leading term in its runtime, i.e., $\sum_{j=1}^{q}\mathrm{nnz}\left(X^{(j)}\right)$. The following fundamental question about whether the factor $q^{1.5}$ in the runtime of (Woodruff & Zandieh, 2020) is necessary has not been answered yet.

Can we produce a sampling matrix that satisfies (1) *in time* $\widetilde{O}\left(\mathrm{poly}(q,\epsilon^{-1})\cdot s_\lambda^2 n + \sum_{j=1}^{q}\mathrm{nnz}\left(X^{(j)}\right)\right)$*?*

We answer the above question positively in Theorem 2.7, which shows that input sparsity runtime and small $s = O(\epsilon^{-2}s_\lambda \log n)$ number of samples are achievable. One advantage of our method is that after computing the sampling matrix Π using Theorem 2.7, we can simply store $\Pi\Phi$ using $O(ns) = O(\epsilon^{-2}s_\lambda n\log n)$ words of memory, while the memory needed to store the exact Gram matrix $\Phi^\top\Phi$ is $\Theta(n^2)$. Thus, our method reduces the memory from quadratic in the dataset size n, to linear.

Additionally, for solving many downstream learning tasks such as ridge regression, low-rank approximation, or PCA with the feature matrix Φ, one typically needs to either compute the inverse or the SVD of the Gram matrix $\Phi^\top\Phi$. If $\Phi^\top\Phi$ is pre-computed exactly and is stored in memory, then computing its SVD requires $\Theta(n^3)$ additional runtime. So the total time to compute $\Phi^\top\Phi$ exactly and then find its SVD, for tensor product feature matrices Φ, is $\Theta\left(n\cdot\sum_{j=1}^{q}\mathrm{nnz}\left(X^{(j)}\right)+n^3\right)$. In contrast, given the sub-sampled feature matrix $\Pi\Phi$, we can (spectrally) approximate the SVD of $\Phi^\top\Phi$ by the SVD of $(\Pi\Phi)^\top(\Pi\Phi)$, using only $s^2 n = O(\epsilon^{-4}s_\lambda^2 n\log^2 n)$ operations. Thus, using our Theorem 2.7, the SVD of $(\Pi\Phi)^\top(\Pi\Phi)$ can be computed in total time $\widetilde{O}\left(\mathrm{poly}(q,\epsilon^{-1})\cdot s_\lambda^2 n + \sum_{j=1}^{q}\mathrm{nnz}\left(X^{(j)}\right)\right)$. Hence, our method improves the runtime of solving downstream applications, such as ridge regression or PCA from cubic in n to linear.

• For the important case when the input datasets are identical $X^{(1)} = X^{(2)} = \cdots X^{(q)} = X$ and the feature matrix $\Phi := X^{\otimes q}$ corresponds to the degree-q polynomial kernel, invoking our Theorem 2.7 results in a runtime of $\widetilde{O}\left(\mathrm{poly}(q,\epsilon^{-1})\cdot s_\lambda^2 n + q\cdot\mathrm{nnz}(X)\right)$, which is a factor q larger than the desired input sparsity time. On the other hand, Song et al. (2021) has recently proposed a linear sketch with $\widetilde{O}(n/\epsilon^2)$ rows which satisfies (1) for $\Phi = X^{\otimes q}$ and can be applied in time $\widetilde{O}\left(q^2\epsilon^{-2}n^2 + nd\right)$, which can be considered to be $\widetilde{O}\left(q^2\epsilon^{-2}n^2 + \mathrm{nnz}(X)\right)$ for dense X, i.e., $\mathrm{nnz}(X) = \widetilde{\Omega}(nd)$. That is, Song et al. (2021) showed that decoupling the factor of q from $\mathrm{nnz}(X)$ is possible at the expense of having sub-optimal target dimension $s \approx n/\epsilon^2$ and losing quadratically in n in the runtime. However, it is unclear whether these losses are necessary. Specifically we consider the following fundamental question:

Can we produce a sampling matrix with $s = O\left(\frac{s_\lambda}{\epsilon^2}\log n\right)$ *rows that satisfies* (1) *for the degree-q polynomial kernel in time* $\widetilde{O}\left(\mathrm{poly}(q,\epsilon^{-1})\cdot s_\lambda^2 n + \mathrm{nnz}(X)\right)$*?*

We answer the above question positively in Theorem 4.3. Specifically, our Theorem 4.3 applies to any matrix $\Phi = X^{\otimes q}$ in time $\widetilde{O}\left(\mathrm{poly}(q,1/\epsilon)\left(s_\lambda^2 + \sqrt{\|K\|/\lambda}\right)\cdot n + dn\right)$, where $K = \Phi^\top\Phi$ is the kernel matrix corresponding to the degree-q polynomial kernel. For large d, this runtime is dominated by $\widetilde{O}(dn)$. Thus, for dense datasets with $\mathrm{nnz}(X) = \widetilde{\Omega}(nd)$, this runtime has the same asymptotic order as the input sparsity $\mathrm{nnz}(X)$, and is thus optimal up to $\log$ factors.

• We generalize our sampling methods to other classes of kernels beyond polynomial, such as the Gaussian and the Neural Tangent Kernels (Jacot et al., 2018) in Section 5. For example in Corollary 5.3, we prove that our sampling method spectrally approximates the Gaussian kernel for dense datasets with squared radius r in time $\widetilde{O}\left(\frac{r^8}{\epsilon^4}s_\lambda^2 n + r^3\sqrt{\frac{n}{\lambda}}n + nd\right)$. For comparison, the runtime of (Song et al., 2021) is $\widetilde{O}\left(\frac{r^3}{\epsilon^2}\cdot n^2 + nd\right)$, which means that for any $\lambda = \omega(1/n)$, any $\epsilon = \widetilde{\Omega}(1)$, and any $r = o\left(n^{0.2}\right)$, the result of our Corollary 5.3 is strictly faster.

• In addition to our theoretical guarantees, we provide regression and classification experiments in Section 6, which show our method performs well in practice even for moderately-sized datasets. In particular, our empirical results show that our method achieves better testing errors compared to prior results for both Gaussian and Neural Tangent kernels.

1.2. Our Techniques

• Our algorithm samples s i.i.d. rows of the feature matrix $\Phi = \bigotimes_{j=1}^{q} X^{(j)}$ according to its ridge leverage scores. We devise a highly optimized version of the recursive sampling framework of (Woodruff & Zandieh, 2020), which previously had a runtime of $\widetilde{O}\left(q^{1.5}\sum_{j=1}^{q}\mathrm{nnz}\left(X^{(j)}\right)\right)$. By closely examining (Woodruff & Zandieh, 2020) we isolate the main computational bottleneck of their algorithm and formulate it as a data-structure (DS) problem in Section 3. In particular, our algorithm crucially relies on an efficient DS that can be constructed in input sparsity time, i.e., $\sum_{j=1}^{q}\mathrm{nnz}\left(X^{(j)}\right)$, and enables estimation of $\left\|\left(\bigotimes_{j=1}^{q} X^{(j)}\right)V\right\|_F^2$ for arbitrary queries $V\in\mathbb{R}^{n\times r}$ in time $\mathrm{poly}(q)\cdot\mathrm{nnz}(V)$. We solve this DS problem in Sec-

tion 3 and then use it in our importance sampling method for tensor product matrices in Section 2.1 and Appendix B.

• To run our sampling algorithm on the feature matrix $X^{\otimes q}$ of the polynomial kernel in input sparsity time, we crucially need a DS that can be constructed in $\mathrm{nnz}(X)$ time and can quickly answer queries of the form $\|X^{\otimes q} \cdot V\|_F^2$. Our main technical tool for solving this problem is a collection of sketches $S^{(1)}, S^{(2)}, \dots S^{(q)}$ which are *correlated* to the extent that they can be simultaneously applied to X in a total of $\widetilde{O}(\mathrm{nnz}(X))$ time, and at the same time are *independent enough* to ensure that $\left\| \left(\bigotimes_{j=1}^{q} S^{(j)} X \right) V \right\|_F^2 \approx \|X^{\otimes q} V\|_F^2$. We show in Section 4.1 that a set of Subsampled Randomized Hadamard Transform (SRHT) sketches with shared random signs can be applied to any dense dataset X in total time $\widetilde{O}(\mathrm{nnz}(X))$, and also provide an unbiased estimator with small variance for $\|X^{\otimes q} V\|_F^2$. It is not clear at this point if variants of sparse sketches (e.g., CountSketch) with these properties also exist or not.

1.3. Related Work

A popular line of work on kernel approximation is based on the Random Fourier Features method (Rahimi & Recht, 2009), which works well for shift-invariant kernels and with some modifications can embed the Gaussian kernel in constant dimension using a near optimal number of features (Avron et al., 2017). However, all variants of this method need at least $\Omega(s_\lambda \cdot \mathrm{nnz}(X))$ runtime which is a factor s_λ higher than our desired time.

Another popular kernel approximation approach is the Nyström method (Williams & Seeger, 2001). While the recursive Nyström sampling of Musco & Musco (2017) can embed kernel matrices using a near optimal number of landmarks, this method also needs at least $\Omega(s_\lambda \cdot \mathrm{nnz}(X))$ runtime, which is a factor s_λ higher than our desired time.

For the polynomial kernel, sketching methods have been developed extensively (Avron et al., 2014; Pham & Pagh, 2013; Woodruff & Zandieh, 2020; Song et al., 2021). For example, Ahle et al. (2020) proposed a subspace embedding for high-degree polynomial kernels as well as the Gaussian kernel. However, their required runtime for the degree q polynomial kernel is at least $\Omega(q \cdot \mathrm{nnz}(X))$, which has an undesirable factor q. Recently, Song et al. (2021) showed that this sketching method can be accelerated for dense datasets by applying an SRHT on the input dataset. However, their resulting runtime is $\widetilde{O}(q^2 n^2 + nd)$ which has an undesirable quadratic dependence on n.

2. Preliminaries

Throughout the paper, we use symbols $e_1, e_2, \dots e_d$ to denote the standard basis vectors in $\mathbb{R}^d$. For any positive integer n, we define the set $[n] = \{1, 2, \dots n\}$. For a matrix A we use $\|A\|$ to denote its operator norm. We also use $A_{i,\star}$ and $A_{\star,i}$ to denote the i^{th} row and i^{th} column of A, respectively. We use the notation $\widetilde{O}(f)$ to denote $O(f \cdot \mathrm{poly}\log f)$, for any f. For any matrix $\Phi \in \mathbb{R}^{m \times n}$ and regularizer $\lambda > 0$, the (row) λ-ridge leverage scores of this matrix are defined as

$$\ell_i^\lambda := \left\| \Phi_{i,\star} (\Phi^\top \Phi + \lambda I)^{-1/2} \right\|_2^2, \text{ for every } i \in [m]. \quad (2)$$

Definition 2.1 (Tensor product)**.** Given $x \in \mathbb{R}^m$ and $y \in \mathbb{R}^n$ we define the tensor product of these vectors as $x \otimes y = xy^\top$. Although tensor products are multidimensional objects, it is convenient to associate them with single-dimensional vectors, so we often associate $x \otimes y$ with $(x_1 y_1, x_2 y_1, \dots x_m y_1, x_1 y_2, \dots x_m y_2, \dots x_m y_n)$.
For shorthand, we use the notation $x^{\otimes p}$ to denote $\underbrace{x \otimes x \otimes \dots x}_{p \text{ terms}}$, the p-fold self-tensoring of x.

We wish to define the column-wise tensoring of matrices as:

Definition 2.2. Given $A^{(1)} \in \mathbb{R}^{m_1 \times n}, \dots, A^{(k)} \in \mathbb{R}^{m_k \times n}$, we define $A^{(1)} \otimes \dots \otimes A^{(k)}$ to be the matrix in $\mathbb{R}^{m_1 \dots m_k \times n}$ whose j^{th} column is $A^{(1)}_{\star,j} \otimes \dots \otimes A^{(k)}_{\star,j}$ for every $j \in [n]$.

A key property of tensor products that we frequently use is that for any matrices A, B, C with a conforming number of columns, there is a bijective correspondence between the elements of $(A \otimes B) \cdot C^\top$ and $A \cdot (B \otimes C)^\top$. More precisely, the entry at row (i, j) and column k of $(A \otimes B) \cdot C^\top$ is equal to the entry at row i and column (j, k) of $A \cdot (B \otimes C)^\top$.

We use a norm-preserving dimensionality reduction technique that can be applied to tensor products in input sparsity time. Specifically, we use the PolySketch transform introduced in (Ahle et al., 2020), which preserves the norms of vectors in $\mathbb{R}^{d^q}$ and can be applied to tensor product vectors $u_1 \otimes u_2 \otimes \dots u_q$ very quickly. The following lemma follows from Theorem 1.1 of (Ahle et al., 2020).

Lemma 2.3 (PolySketch)**.** *For every positive integers q, d, and every $\epsilon > 0$, there exists a distribution on random matrices $S^q \in \mathbb{R}^{m \times d^q}$ with $m = O\left(\frac{q}{\epsilon^2}\right)$, called* degree-$q$ PolySketch *, such that,*

1. $\Pr\left[\|S^q Y\|_F^2 \in (1 \pm \epsilon)\|Y\|_F^2\right] \geq 19/20$ *for any* $Y \in \mathbb{R}^{d^q \times n}$.

2. For any vectors $u_1, u_2, \dots u_q \in \mathbb{R}^d$, the total time to compute $S^q\left(e_1^{\otimes j} \otimes u_{j+1} \otimes u_{j+2} \otimes \dots u_q\right)$ for all $j = 0, 1, \dots q$ is $O\left(\frac{q^2 \log^2 \frac{q}{\epsilon}}{\epsilon^2} + \sum_{j=1}^{q} \mathrm{nnz}\,(u_j)\right)$.

For a proof of Lemma 2.3, see Appendix A. We also use the Subsampled Randomized Hadamard Transform (SRHT) (Ailon & Chazelle, 2009), which is a norm-preserving dimensionality reduction with near linear runtime.

Algorithm 1 RECURSIVE LEVERAGE SCORE SAMPLING

input: Matrix $\Phi \in \mathbb{R}^{m\times n}$ and $\lambda, \epsilon, \mu > 0$
output: Sampling matrix $\Pi \in \mathbb{R}^{s\times m}$

$s \leftarrow C\frac{\mu}{\epsilon^2}\log_2 n$ for some constant C
$\Pi_0 \leftarrow \{0\}^{1\times m}$, $\lambda_0 \leftarrow \|\Phi\|_F^2/\epsilon$ and $T \leftarrow \log_2 \frac{\lambda_0}{\lambda}$
for $t = 1$ to T **do**
  $\Pi_t \leftarrow$ ROWSAMPLER $(\Phi, \Pi_{t-1}\Phi, \lambda_{t-1}, s)$
  $\lambda_t \leftarrow \lambda_{t-1}/2$
return Π_T

Lemma 2.4 (SRHT Sketch)**.** *For every positive integer d and every $\epsilon, \delta > 0$, there exists a distribution on random matrices $S \in \mathbb{R}^{m\times d}$ with $m = O\left(\frac{1}{\epsilon^2}\cdot\log\frac{1}{\epsilon\delta}\log\frac{1}{\delta}\right)$, called* SRHT *, such that for any matrix $X \in \mathbb{R}^{d\times n}$, $\Pr\left[\|SX\|_F^2 \in (1\pm\epsilon)\|X\|_F^2\right] \geq 1-\delta$. Moreover, SX can be computed in time $O\left(mn + nd\log d\right)$.*

2.1. Recursive Leverage Score Sampling for $\bigotimes_{j=1}^q X^{(j)}$

Algorithm 1 is a generic procedure for sampling the rows of a matrix $\Phi \in \mathbb{R}^{m\times n}$ with probabilities proportional to their leverage scores, restated from (Woodruff & Zandieh, 2020). It starts by generating samples from a crude approximation to the leverage score distribution and then iteratively refines the distribution. The core primitive used in Algorithm 1 is ROWSAMPLER, which samples rows of a certain matrix with probabilities proportional to their squared norms.

A *row norm sampler* is defined in (Woodruff & Zandieh, 2020) as follows,

Definition 2.5 (Row Norm Sampler)**.** Let Φ be an $m \times n$ matrix and s be some positive integer. A *rank-s row norm sampler* for Φ is a random matrix $S \in \mathbb{R}^{s\times m}$ which is constructed by first generating s i.i.d. samples $j_1, j_2, \cdots j_s \in [m]$ from some distribution $\{p_i\}_{i=1}^m$ which satisfies $p_i \geq \frac{1}{4}\frac{\|\phi_{i,\star}\|_2^2}{\|\Phi\|_F^2}$ for all $i \in [m]$, and then letting the r^{th} row of S be $\frac{1}{\sqrt{s\cdot p_{j_r}}}e_{j_r}^\top$ for every $r \in [s]$.

Now we restate the correctness guarantee of Algorithm 1 from (Woodruff & Zandieh, 2020).

Lemma 2.6. *Suppose for any matrices $\Phi \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{r\times n}$, any $\lambda' > 0$, and integer $s > 0$, the primitive* ROWSAMPLER(Φ, B, λ', s) *returns a rank-s row norm sampler for $\Phi(B^\top B + \lambda' I)^{-1/2}$ as in Definition 2.5. Then for any $\lambda, \epsilon > 0$, any $\Phi \in \mathbb{R}^{m\times n}$ with statistical dimension $s_\lambda = \|\Phi(\Phi^\top\Phi + \lambda I)^{-1/2}\|_F^2$, and $\mu \geq s_\lambda$, Algorithm 1 returns a sampling matrix $\Pi \in \mathbb{R}^{s^*\times m}$ with $s^* = O(\frac{\mu}{\epsilon^2}\log n)$ rows such that with probability $1 - \frac{1}{\text{poly}(n)}$, $\Phi^\top\Pi^\top\Pi\Phi$ is an (ϵ, λ)-spectral approximation to $\Phi^\top\Phi$ as per (1).*

Given this lemma, our goal is to run Algorithm 1 on $\Phi = \bigotimes_{j=1}^q X^{(j)}$ in nearly $\sum_i \text{nnz}\left(X^{(i)}\right)$ time. This crucially requires an efficient implementation of ROWSAMPLER, which carries out the main computations. We show in Appendix B that there exists an efficient ROWSAMPLER primitive for matrices of the form $\Phi(B^\top B + \lambda I)^{-1/2}$, for any B. Our algorithm employs a data-structure for efficient estimation of queries of the form $\left\|\left(\bigotimes_{j=1}^q X^{(j)}\right)V\right\|_F^2$, which we will design and present in Section 3, and heavily exploits various properties of tensor products. See Algorithm 4 and Lemma B.1 for details. We further prove the following main theorem in Appendix B.

Algorithm 2 DS for estimating $\left\|\left(\bigotimes_{j=1}^q X^{(j)}\right)V\right\|_F^2$

input: Matrices $X^{(1)}, \ldots X^{(q)} \in \mathbb{R}^{d\times n}$, $\varepsilon > 0$

$m \leftarrow C_1\frac{q}{\varepsilon^2}$, $T \leftarrow C_2\log n$, $m' \leftarrow C_3\frac{\log(1/\varepsilon)}{\varepsilon^2}$
For every $i \in [T]$, let $Q_i \in \mathbb{R}^{m'\times m}$ be independent copies of the SRHT as per Lemma 2.4, and let $S_i^q \in \mathbb{R}^{m\times d^q}$ be independent copies of the degree-q POLYSKETCH as per Lemma 2.3
Compute $P_{i,j} \leftarrow Q_i \cdot S_i^q\left(E_1^{\otimes j}\otimes X^{(j+1)}\otimes\ldots X^{(q)}\right)$, for every $i \in [T]$ and $j = 0, 1, \ldots q$, where $E_1 \in \mathbb{R}^{d\times n}$ is defined as $E_1 := [e_1, e_1, \ldots e_1]$

Procedure QUERY (V, j)

$\tilde{z}_j \leftarrow$ MEDIAN$_{i\in[T]}\left\{\|P_{i,j}\cdot V\|_F^2\right\}$
return $\tilde{z}_j$

Theorem 2.7. *For any collection of matrices $X^{(1)}, X^{(2)}, \ldots X^{(q)} \in \mathbb{R}^{d\times n}$ and any $\epsilon, \lambda > 0$, if matrix $\Phi := \bigotimes_{j=1}^q X^{(j)}$ has statistical dimension $s_\lambda = \|\Phi(\Phi^\top\Phi + \lambda I)^{-1/2}\|_F^2$ and $\frac{\|\Phi\|_F^2}{\epsilon\lambda} \leq \text{poly}(n)$, then there exists an algorithm that returns a random sampling matrix $\Pi \in \mathbb{R}^{s\times d^q}$ with $s = O(\frac{s_\lambda}{\epsilon^2}\log n)$ in time $O\left(\text{poly}(q, \log n, \epsilon^{-1})\cdot s_\lambda^2 n + \log^4 n\log q\sum_i nnz\left(X^{(i)}\right)\right)$ such that with probability $1 - \frac{1}{\text{poly}(n)}$, $\Phi^\top\Pi^\top\Pi\Phi$ is an (ϵ, λ)-spectral approximation to $\Phi^\top\Phi$ as per (1).*

3. Data Structure for Estimating $\left\|\left(\bigotimes_{j=1}^q X^{(j)}\right)V\right\|_F^2$

At the core of our leverage score sampling algorithm, we have a new data-structure (DS) that can efficiently answer queries of the form $\left\|\left(\bigotimes_{j=1}^q X^{(j)}\right)V\right\|_F^2$. In this section, we solve the following DS problem,

TENSORNORM DS Problem. For every matrices $X^{(1)}, X^{(2)}, \ldots X^{(q)} \in \mathbb{R}^{d\times n}$ and every $\epsilon > 0$, we want to design a DS called TENSORNORMDS such that,

- The time to construct TENSORNORMDS and the memory needed to store it are $\widetilde{O}\left(\sum_{j=1}^q \text{nnz}\left(X^{(j)}\right)\right)$ and

$\widetilde{O}\left(\text{poly}(q,\epsilon^{-1})\cdot n\right)$, respectively.

- There exists an algorithm that, given TENSORNORMDS and every query $V\in\mathbb{R}^{n\times r}$ and $j=0,\dots q-1$, outputs an estimator $\tilde{z}_j$ in time $\widetilde{O}\left(\text{poly}(q,\epsilon^{-1})\cdot \text{nnz}(V)\right)$, such that,

$$\tilde{z}_j\in(1\pm\epsilon)\left\|\left(X^{(j+1)}\otimes\dots X^{(q)}\right)V\right\|_F^2. \tag{3}$$

Using POLYSKETCH and SRHT , we design TENSORNORMDS in Algorithm 2 and analyze it in the following lemma.

Lemma 3.1 (TensorNorm Data-structure)**.** *For any input datasets $X^{(1)},X^{(2)},\dots X^{(q)}\in\mathbb{R}^{d\times n}$ and any $\epsilon>0$, Algorithm 2 constructs a DS such that given this DS, the procedure* QUERY(V,j)*, for any query $V\in\mathbb{R}^{n\times r}$ and $j=0,1,\dots q$, outputs $\tilde{z}_j$ that satisfies (3) with probability $1-\frac{1}{\text{poly}(n)}$. The time to construct the DS is $O\left(\frac{q^2\log^2\frac{q}{\epsilon}}{\epsilon^2}\cdot n\log n+\log n\cdot\sum_{j=1}^q\text{nnz}\left(X^{(j)}\right)\right)$. Additionally, the memory required to store this DS and the runtime of* QUERY(V,j) *are $O\left(\frac{q\log(1/\epsilon)}{\epsilon^2}n\log n\right)$ and $O\left(\frac{\log(1/\epsilon)}{\epsilon^2}\log n\cdot\text{nnz}(V)\right)$, respectively.*

We prove this lemma in Appendix A.1. Given this DS and using Algorithm 1, we can generate leverage score samples for $\Phi=\bigotimes_{j=1}^q X^{(j)}$.

4. High Degree Polynomial Kernels

Using Theorem 2.7, one can spectrally approximate the Gram matrix of a degree-q self tensor product $X^{\otimes q}$, in time $\widetilde{O}\left(\text{poly}(q,\epsilon^{-1})\cdot s_\lambda^2 n+q\cdot\text{nnz}(X)\right)$. Note that $X^{\otimes q\top}X^{\otimes q}$ is in fact the kernel matrix corresponding to the degree-q polynomial kernel. While this is fast, it is still a factor of q slower than our desired input sparsity runtime (i.e., fastest achievable runtime). We want to understand the following fundamental question:

Is the factor q in runtime necessary, or can one achieve a runtime of $\widetilde{O}(\text{nnz}(X))$?

We show that it is possible to shave off the factor q and achieve $\widetilde{O}(\text{nnz}(X))$ time complexity, at least for dense datasets X. Our main technical tool is a new variant of SRHT sketches that are partially correlated by sharing the same random signs.

4.1. SRHT Sketches with Shared Random Signs

Consider the DS problem in Section 3 for a self-tensor product matrix $X^{\otimes q}$. To estimate $\|X^{\otimes q}\cdot V\|_F^2$ for query matrices V, we can use TENSORNORMDS (Algorithm 2); however, the time to construct this DS is $\widetilde{O}\left(q\cdot\text{nnz}(X)\right)$, by Lemma 3.1. Our goal is to improve this runtime by a factor of q and be able to construct this DS in input sparsity time.

A natural approach for doing so is to first apply a linear sketch, say S, on the dataset X to reduce its size (number of rows) and then construct TENSORNORMDS for $(SX)^{\otimes q}$. To make this work, one needs to ensure that the sketch S satisfies $\|(SX)^{\otimes q}V\|_F^2\approx\|X^{\otimes q}V\|_F^2$ for every query V (at least with constant probability). One way of ensuring this condition, as shown in (Song et al., 2021, Lemma 4.5), is through requiring S to satisfy the oblivious subspace embedding (OSE) property. However, this would require S to have at least n rows, which results in an undesirable quadratic in n running time (recall that our aim is to have a linear in n runtime for constructing the DS).

On the other hand, an OSE might seem like overkill because we just want to estimate $\|X^{\otimes q}V\|_F^2$ for some fixed queries V. One might hope that the weaker JL property would be sufficient for S. However, this is not the case. To see why, suppose for simplicity that $q=2$. Also let v be the all ones vector in $\mathbb{R}^n$ i.e., $v=\mathbf{1}_n$, and let $X\in\mathbb{R}^{d\times n}$ have orthonormal rows. By basic properties of tensor products we have $\left\|X^{\otimes 2}\cdot v\right\|_2^2=d$ and our estimator is $\left\|(SX)^{\otimes 2}\cdot v\right\|_2^2=\left\|SX\cdot\text{diag}(v)\cdot X^\top S^\top\right\|_F^2=\left\|SS^\top\right\|_F^2$. Now if S, for instance, is a random Gaussian matrix, $\left\|SS^\top\right\|_F^2$ is not even an unbiased estimator and has a large bias, i.e., $\mathbb{E}\left[\left\|SS^\top\right\|_F^2\right]\neq\left\|X^{\otimes 2}\cdot v\right\|_2^2=d$. It is not clear at all that a Gaussian matrix with a small $\text{poly}(\log n)$ number of rows would be sufficient to have $\left\|SS^\top\right\|_F^2\approx d$. Note that Sparse JL transforms have even larger variance and bias than Gaussian sketches. The main issue here is the fact that we used a single sketch matrix.

If we had independent JL transforms, S_1 and S_2, then $\left\|S_1S_2^\top\right\|_F^2$ would be a good estimator for $\left\|X^{\otimes 2}\cdot v\right\|_2^2=d$. However, using two identical copies of a single sketch introduces dependencies that are problematic even in the toy example of $q=2$.

Thus, we need to construct a collection of sketches $S^{(1)},S^{(2)},\dots S^{(q)}$ which are *correlated* to the extent that would make computation of $S^{(j)}X$ in total time $\widetilde{O}(\text{nnz}(X))$ possible, and at the same time are *independent enough* to ensure that $\left\|\left(\bigotimes_{j=1}^q S^{(j)}X\right)V\right\|_F^2\approx\|X^{\otimes q}V\|_F^2$ while the number of rows of the sketches is small. We achieve this by using a set of correlated SRHT sketches that can be simultaneously applied to X in a total runtime that only depends on the size of the dataset X. We prove that for a collection of SRHT 's with shared random signs, the sketched matrices $S^{(j)}X$ can be computed quickly and $\left\|\left(\bigotimes_{j=1}^q S^{(j)}X\right)V\right\|_F^2$ is an unbiased estimator for $\|X^{\otimes q}V\|_F^2$ with a small variance. It is not clear at this point if variants of sparse sketches (e.g., CountSketch) with these properties exist or not.

Furthermore, note that the eventual use of the DS for estimating $\|X^{\otimes q}V\|_F^2$ will be in our sampling method in Section 4.2 and as it turns out, the queries V that our sampling algorithm produces exhibit some structure. We exploit these structures to prove tighter norm estimation bounds for our new family of correlated SRHT 's in the following lemma.

Lemma 4.1 (SRHT Sketches with Shared Random Signs)**.** *Let $D \in \mathbb{R}^d$ be a diagonal matrix with i.i.d. Rademacher diagonal entries and let $H \in \mathbb{R}^{d\times d}$ be the Hadamard matrix and also let $P_1, P_2, \dots P_q \in \mathbb{R}^{m\times d}$ be independent random sampling matrices that sample m random coordinates of $\mathbb{R}^d$. Define the collection of SRHT sketches with shared signs $\left(S^{(1)}, S^{(2)}, \dots S^{(q)}\right)$ as $S^{(c)} := \frac{1}{\sqrt{m}} \cdot P_c H D$ for $c \in [q]$. For any $X \in \mathbb{R}^{d\times n}$, any PSD matrix $K \in \mathbb{R}^{n\times n}$ with condition number $\kappa := \frac{\lambda_{\max}(K)}{\lambda_{\min}(K)}$, any matrix $\Sigma \in \mathbb{R}^{d'\times n}$, and any $\epsilon, \delta > 0$, if $m = \Omega\left(\left(\frac{1}{\epsilon^2} + \frac{\kappa}{\epsilon}\right) \cdot \frac{q}{\delta}\log n\right)$, then with probability at least $1-\delta$,*

$$\left\| \left[\bigotimes_{c=1}^{q} S^{(c)} X\right] (\Sigma \otimes K)^\top \right\|_F^2 \in (1\pm\epsilon) \left\| X^{\otimes q} (\Sigma \otimes K)^\top \right\|_F^2$$

Furthermore, the total time to compute $S^{(1)}X, \dots S^{(q)}X$ is bounded by $O\left(qmn + nd\log d\right)$.

We prove this lemma in Appendix C. According to Lemma 4.1, the Kronecker product of SRHT sketches with shared random signs $S^{(1)} \times S^{(2)} \times \dots S^{(q)}$ acts as a near-isometry for matrices of the form $X^{\otimes q} \cdot (\Sigma \otimes K)^\top$ with constant probability, as long as the target dimension of the $S^{(c)}$'s is at least $m \approx (\epsilon^{-2} + \epsilon^{-1}\kappa) q \log n$. If the $S^{(c)}$ sketches were fully independent, as in (Ahle et al., 2020), then a target dimension of $m \approx \epsilon^{-2} q \log n$ would suffice. So the price of using correlated sketches is a factor of $\epsilon\kappa + 1$ increase in the target dimension. On the other hand, letting the sketches $S^{(c)}$ use independent sampling matrices is critical. If we used identical SRHT 's $S^{(1)} = \dots S^{(q)} = S$, as is done in Lemma 4.5 of (Song et al., 2021), then to have the guarantee of Lemma 4.1, the sketch S would need to be an OSE, which requires a target dimension of $m = \Omega\left(\frac{q^2}{\epsilon^2} \cdot n \log n\right)$. Lemma 4.1 provides a target dimension improvement over the OSE-based results by a factor of $\frac{qn}{1+\epsilon\kappa}$, which is significant.

Lemma 4.1 shows us a way of speeding up the DS given in Algorithm 2 for self tensor products $X^{\otimes q}$. One can quickly compute sketched datasets $Y^{(r)} = S^{(r)}X$ for every $r \in [q]$, and then apply TENSORNORMDS to $Y^{(1)}, \dots Y^{(q)}$, in total time $\widetilde{O}(\mathrm{nnz}(X))$ for dense X. It turns out that all queries that our sampling algorithm in Section 4.2 produces are exactly of the form $V = (\Sigma \otimes K)^\top$. Thus, the combination of Lemma 4.1 and Algorithm 2 is a perfect solution for our sampling algorithm's norm estimation needs.

Algorithm 3 ROWSAMPLER for $X^{\otimes q}$

input: $q, s \in \mathbb{Z}_+$, $X \in \mathbb{R}^{d\times n}$, $B \in \mathbb{R}^{m\times n}$, $\lambda > 0$
output: Sampling matrix $S \in \mathbb{R}^{s\times d^q}$

$\kappa \leftarrow \sqrt{\frac{\|B^\top B\|}{\lambda} + 1}$
Generate $H \in \mathbb{R}^{d'\times n}$ with i.i.d. normal entries with $d' = C_0 q^2 \log n$ rows
$M \leftarrow H \cdot (B^\top B + \lambda I)^{-1/2}$
For every $k \in [m']$, let $S_k^{(1)}, S_k^{(2)}, \dots S_k^{(q)} \in \mathbb{R}^{m''\times d}$ be independent copies of SRHT sketches with shared signs as per Lemma 4.1, where $m' = C_1 \log n$ and $m'' = C_2(q^3 + q^2\kappa)\log n$
For every $k \in [m']$, let $\mathrm{TN}^{(k)}$ be the DS in Algorithm 2 for inputs $\left(S_k^{(1)}X, \dots S_k^{(q)}X, M\right)$ and $\epsilon = \frac{1}{40q}$
Let $h : [d] \to [s']$ be a fully independent and uniform hash function with $s' = \lceil q^3 s \rceil$ buckets
Let $h^{-1}(r) = \{j \in [d] : h(j) = r\}$ for every $r \in [s']$
For every $r \in [s']$ and $k \in [m']$, lget $G_r^k \in \mathbb{R}^{n'\times d_r}$ be independent instances of degree-1 POLYSKETCH as per Lemma 2.3, where $d_r = |h^{-1}(r)|$, $n' = C_3 q^2$
$W_{r,k} \leftarrow G_r^k \cdot X_{h^{-1}(r),\star}$ for every $k \in [m']$ and $r \in [s']$
for $\ell = 1$ to s **do**
  $D^1 \leftarrow I_n$ and $\beta_\ell \leftarrow s$
  for $a = 1$ to q **do**
    $L_{r,k}^a \leftarrow D^a \cdot W_{r,k}^\top$ for every $k \in [m']$, and $r \in [s']$
    $p_r^a \leftarrow \mathrm{MEDIAN}_{k\in[m']} \mathrm{TN}^{(k)}.\mathrm{QUERY}(L_{r,k}^a, a)$ for every $r \in [s']$
    $p_r^a \leftarrow p_r^a / \sum_{t=1}^{s'} p_t^a$ for every $r \in [s']$
    Sample $t \in [s']$ from distribution $\{p_r^a\}_{r=1}^{s'}$
    Let $q_i^a \leftarrow \mathrm{MEDIAN}_{k\in[m']} \mathrm{TN}^{(k)}.\mathrm{QUERY}(D^a X_{i,\star}^\top, a)$ for every $i \in h^{-1}(t)$
    $q_i^a \leftarrow q_i^a / \sum_{j\in h^{-1}(t)} q_j^a$ for every $i \in h^{-1}(t)$
    Sample $i_a \in [d]$ from distribution $\{q_i^a\}_{i\in h^{-1}(t)}$
    $D^{a+1} \leftarrow D^a \cdot \mathrm{diag}\left(X_{i_a,\star}^{(a)}\right)$
    $\beta_\ell \leftarrow \beta_\ell \cdot p_t^a q_{i_a}^a$
  Let ℓ^{th} row of S be $\beta_\ell^{-1/2} \left(e_{i_1} \otimes e_{i_2} \otimes \cdots e_{i_q}\right)^\top$
return S

4.2. ROWSAMPLER for Degree-q Self-Tensor Products

In this section, we design an algorithm that can perform row norm sampling (see Definition 2.5) on a matrix of the form $X^{\otimes q}(B^\top B + \lambda I)^{-1/2}$ using $\widetilde{O}(\mathrm{nnz}(X))$ runtime for dense X. Our primitive crucially relies on TENSORNORMDS (Algorithm 3) as well as our new variant of SRHT with shared random signs that we analyzed in Lemma 4.1.

Overview of Algorithm 3: The goal of ROWSAMPLER is to generate samples $(i_1, i_2, \cdots i_q) \in [d]^q$ with probabilities proportional to the squared norm of the row $(i_1, \cdots i_q)$ of matrix $X^{\otimes q}(B^\top B + \lambda I)^{-1/2}$. Because $(B^\top B + \lambda I)^{-1/2}$

has a large $n \times n$ size, we first compress it without perturbing the distribution of row norms of $X^{\otimes q}(B^\top B + \lambda I)^{-1/2}$ too much. This can be done by applying a JL-transformation to the rows of this matrix (see, e.g., (Dasgupta & Gupta, 2003)). Let $H \in \mathbb{R}^{d' \times n}$ be a random matrix with i.i.d. normal entries with $d' = C_0 q^2 \log_2 n$ rows. With probability $1 - \frac{1}{\text{poly}(n)}$ the norm of each row of the sketched matrix $X^{\otimes q}(B^\top B + \lambda I)^{-1/2} \cdot H^\top$ is preserved up to a $\left(1 \pm O(q^{-1})\right)$ factor. This is done in line 3 of the algorithm by computing $M := H \cdot (B^\top B + \lambda I)^{-1/2}$, which can be computed quickly since B and H have a small number of rows.

Now the problem is reduced to performing row norm sampling on $X^{\otimes q} M^\top$. Note that computing the exact row norms of this matrix is out of the question since it has a huge d^q number of rows. However, by using TENSORNORMDS that we designed in Algorithm 2 and the new variant of SRHT sketches we introduced in Lemma 4.1 and by exploiting properties of tensor products we can generate samples from the row norm distribution as follows.

By basic properties of tensor products, the entries of $X^{\otimes q} M^\top$ are in bijective correspondence with the entries of $X^{\otimes (q-1)} \cdot (X \otimes M)^\top$, where the entry at row $(i_1, i_2, \cdots i_q)$ and column j of $X^{\otimes q} M^\top$ is equal to the entry at row $(i_2, \dots i_q)$ and column (i_1, j) of $X^{\otimes (q-1)} \cdot (X \otimes M)^\top$.

Therefore, it is enough to have a procedure to sample $(i_1, i_2, \dots i_q)$ with probability proportional to the squared norm of the row $(i_2, \dots i_q)$ of matrix $X^{\otimes (q-1)} \cdot (M \cdot \text{diag}\,(X_{i_1,\star}))^\top$ for every $i_1 \in [d]$. We do this task in two steps; first we sample an index i_1 with probability proportional to the squared Frobenius norm of $X^{\otimes (q-1)} \cdot (M \cdot \text{diag}\,(X_{i_1,\star}))^\top$, and then we perform row norm sampling on the sampled matrix.

To do the first sampling step above, we need to cheaply estimate the Frobenius norms of matrices $X^{\otimes (q-1)} \cdot (M \cdot \text{diag}\,(X_{i_1,\star}))^\top$. We can estimate such norms using TENSORNORMDS given in Algorithm 2. However, note that $\sum_{j=1}^{q-1} \widetilde{O}(\text{nnz}(X)) = \widetilde{O}(q \cdot \text{nnz}(X))$ operations are required to build this DS. This is where the SRHT sketches with shared random signs plays an important role. If we let $S^{(1)}, \dots S^{(q)} \in \mathbb{R}^{m'' \times d}$ be the SRHT sketches with shared signs as per Lemma 4.1, then we can compute $S^{(c)} X$ for all $c \in [q]$ in time $O(nd \log d) = \widetilde{O}(\text{nnz}(X))$, for dense datasets X. Now we can cheaply estimate the Frobenius norms of matrices $\left(\bigotimes_{c=1}^{q-1} S^{(c)} X\right) \cdot (M \cdot \text{diag}\,(X_{i_1,\star}))^\top$ up to a small perturbation using TENSORNORMDS (Algorithm 2) because the sketched matrices $S^{(c)} X$ have small sizes. We let the target dimension of these sketches be $m'' = C_2(q^3 + q^2\kappa) \log n$, where $\kappa = \sqrt{\frac{\|B^\top B\|}{\lambda} + 1}$ is the condition number of $(B^\top B + \lambda I)^{-1/2}$. Thus, by Lemma 4.1 and using the fact that matrix $M = H(B^\top B + \lambda I)^{-1/2}$ for a JL matrix H, the Frobenius norm of $\left(\bigotimes_{c=1}^{q-1} S^{(c)} X\right) \cdot (M \cdot \text{diag}\,(X_{i_1,\star}))^\top$ is within a factor $(1 \pm O(q^{-1}))$ of the Frobenius norm of $X^{\otimes (q-1)} \left((B^\top B + \lambda I) \cdot \text{diag}\,(X_{i_1,\star})\right)^\top$.

After this point, we will have an index $i_1 \in [d]$ sampled from the correct distribution and all that is left to do is to carry out row norm sampling on $X^{\otimes (q-1)} \left(M \cdot \text{diag}\left(X^{(1)}_{i_1,\star}\right)\right)^\top$. Note that we have made progress because this matrix has d^{q-1} rows, so we have reduced the size of our problem by a factor of d. Algorithm 4 recursively repeats this process of reshaping and sketching and sampling with the aid of our DS, q times until having all q indices $i_1, i_2, \cdots i_q$. Note that the actual procedure requires more work because we need to generate s i.i.d. samples from the distribution of row norms, and in order to ensure that the runtime does not lose a multiplicative factor of s, resulting in $s \cdot \text{nnz}(X)$ total time, we need to perform additional sketching and a random partitioning of the rows of the datasets to $q^3 s$ buckets. We also boost the success probability of all these operations, when necessary, using the median trick.

The formal guarantee on Algorithm 3 is given in the following lemma.

Lemma 4.2. *For any matrix $X \in \mathbb{R}^{d \times n}$ and $B \in \mathbb{R}^{m \times n}$, any $\lambda > 0$ and any positive integers q, s, with probability at least $1 - \frac{1}{\text{poly}(n)}$, Algorithm 3 outputs a rank-s row norm sampler for $X^{\otimes q}(B^\top B + \lambda I)^{-1/2}$ as per Definition 2.5 in time $O\left(m^2 n + q^8 s^2 n \log^3 n + q^3 \kappa n \log^3 n + nd \log^4 n\right)$, where $\kappa = \sqrt{\|B^\top B\|/\lambda + 1}$.*

We prove Lemma 4.2 in Appendix D.1. Now we can give our main theorem about spectrally approximating the degree-q polynomial kernel matrix $X^{\otimes q \top} X^{\otimes q}$ using nearly $\text{nnz}(X)$ runtime for dense datasets.

Theorem 4.3. *For any dataset $X \in \mathbb{R}^{d \times n}$ and any $\epsilon, \lambda > 0$, if matrix $\Phi := X^{\otimes q}$ has statistical dimension $s_\lambda = \|\Phi(\Phi^\top \Phi + \lambda I)^{-1/2}\|_F^2$ and $\frac{\|\Phi\|_F^2}{\epsilon \lambda} \leq \text{poly}(n)$, then there exists an algorithm that returns a random sampling matrix $\Pi \in \mathbb{R}^{s \times d^q}$ with sampling dimension $s = O(\frac{s_\lambda}{\epsilon^2} \log n)$ in time $O\left(\frac{q^8 s_\lambda^2 n \log^5 n}{\epsilon^4} + \sqrt{\frac{\|\Phi^\top \Phi\|}{\lambda}} q^3 n \log^3 n + nd \log^5 n\right)$ such that with probability $1 - \frac{1}{\text{poly}(n)}$, $\Phi^\top \Pi^\top \Pi \Phi$ is an (ϵ, λ)-spectral approximation to $\Phi^\top \Phi$ as per (1).*

For a proof of this theorem see Appendix D.2.

Remark on the runtime of Theorem 4.3. Assuming that $\frac{\|\Phi^\top \Phi\|}{\lambda} \leq \text{poly}(q/\epsilon) \cdot s_\lambda^4$, the low order term of our algorithm's runtime is $\widetilde{O}(\text{poly}(q/\epsilon) \cdot s_\lambda^2 n)$. While the quadratic dependence on s_λ might seem like a limitation, we argue that for a wide range of downstream applications this is not

an issue. In particular, for applications such as regression or PCA, one needs to either invert or compute the SVD of the approximated Gram matrix $(\Pi\Phi)^\top(\Pi\Phi)$ and both of these operations require s^2n runtime, where s is the target dimension of the matrix Π. Note that for any method to achieve the spectral approximation guarantee of (1), the target dimension has to be at least $s = \Omega(s_\lambda)$ (Avron et al., 2019). Thus, the runtime of solving the mentioned downstream learning tasks using any sketching or sampling method is at least $\Omega(s_\lambda^2 n)$, which shows that quadratic dependence on s_λ is unavoidable. For comparison against prior results note that, the sketch in (Song et al., 2021) has a target dimension of $m \approx n/\epsilon^2$. Thus, the total time of using their algorithm to approximately solve kernel ridge regression (KRR) or PCA is $\Theta(n^3/\epsilon^4 + q^2n^2/\epsilon^2 + dn)$.

5. Generalization to Other Kernels

In this section we generalize our sampling algorithms to other classes of kernels such as Gaussian, dot-product, and Neural Tangent kernels. We start by defining a class of kernels that encompasses all aforementioned kernels,

Definition 5.1 (Generalized Polynomial Kernel)**.** Given a positive integer q, a vector of coefficients $\alpha \in \mathbb{R}^{q+1}$, a vector $v \in \mathbb{R}^n$, and a dataset $X \in \mathbb{R}^{d\times n}$, we define the corresponding *generalized polynomial kernel (GPK)* matrix $K \in \mathbb{R}^{n\times n}$ as $K := \mathrm{diag}(v)\left(\sum_{j=0}^q \alpha_j^2 \cdot X^{\otimes j\top} X^{\otimes j}\right)\mathrm{diag}(v)$. The GPK matrix can be expressed as a Gram matrix $K = \Phi^\top\Phi$ for

$$\Phi := \bigoplus_{j=0}^{q} \alpha_j X^{\otimes j} \cdot \mathrm{diag}(v). \tag{4}$$

We show in Appendix E, how to adapt our leverage score sampling method to the GPK feature matrix Φ defined in (4) and prove the following main theorem,

Theorem 5.2. *Let $\Phi \in \mathbb{R}^{m\times n}$ and K be the GPK feature matrix and kernel matrix defined in Definition 5.1. For any $\epsilon, \lambda > 0$, if Φ has statistical dimension $s_\lambda = \|\Phi(K+\lambda I)^{-1/2}\|_F^2$ and $\frac{\|\Phi\|_F^2}{\epsilon\lambda} \le \mathrm{poly}(n)$, then there exists an algorithm that returns a random sampling matrix $\Pi \in \mathbb{R}^{s\times m}$ with $s = O(\frac{s_\lambda}{\epsilon^2}\log n)$ rows in time $O\left(\frac{q^8 s_\lambda^2 n \log^5 n}{\epsilon^4} + \sqrt{\frac{\|K\|}{\lambda}} q^3 n \log^3 n + nd\log^5 n\right)$ such that with probability $1 - \frac{1}{\mathrm{poly}(n)}$, $\Phi^\top\Pi^\top\Pi\Phi$ is an (ϵ,λ)-spectral approximation to K as per (1).*

Gaussian Kernel. We show in Appendix E.1 that the class of GPK kernels contains a good approximation to the Gaussian kernel matrix for datasets with bounded ℓ_2 norm and therefore, we have the following corollary of Theorem 5.2:

Corollary 5.3 (Application to Gaussian Kernel)**.** *For any $r > 0$ and dataset $x_1, \dots x_n \in \mathbb{R}^n$ with $\max_{i\in[n]}\|x_i\|_2^2 \le r$, any $\lambda, \epsilon > 0$, if $K \in \mathbb{R}^{n\times n}$ is the Gaussian kernel matrix, i.e., $K_{i,j} := e^{-\|x_i - x_j\|_2^2/2}$, with statistical dimension $s_\lambda = \mathrm{tr}\left(K(K+\lambda I)^{-1}\right)$, then there exists an algorithm that computes $Z \in \mathbb{R}^{s\times n}$ with $s = O(\frac{s_\lambda}{\epsilon^2}\log n)$ in time $\widetilde{O}\left(\frac{r^8}{\epsilon^4}s_\lambda^2 n + r^3\sqrt{\frac{\|K\|}{\lambda}}n + nd\right)$ such that with probability $1 - \frac{1}{\mathrm{poly}(n)}$, $Z^\top Z$ is an (ϵ,λ)-spectral approximation to K.*

Note that for the Gaussian kernel we have $\|K\| \le \mathrm{tr}(K) = n$. Therefore, for constant ϵ, the runtime of Corollary 5.3 is always upper bounded by $\widetilde{O}\left(r^8 s_\lambda^2 n + r^3\sqrt{\frac{n}{\lambda}}\cdot n + nd\right)$. For comparison, the runtime of (Song et al., 2021) for spectrally approximating the Gaussian kernel matrix is $\widetilde{O}\left(r^3\cdot n^2 + nd\right)$, which means that for any $\lambda = \omega(1/n)$ and any $r = o\left(n^{0.2}\right)$, our runtime is strictly faster than the runtime of (Song et al., 2021).

Neural Tangent Kernel (NTK). We consider the NTK corresponding to an infinitely wide neural network with two layers and ReLU activation function. This kernel function is defined as follows for any $x, y \in \mathbb{R}^d$ (Zandieh et al., 2021)

$$\Theta_{\mathtt{ntk}}(x,y) := \|x\|_2\|y\|_2 \cdot k_{\mathtt{ntk}}\left(\frac{\langle x,y\rangle}{\|x\|_2\|y\|_2}\right), \tag{5}$$

$$k_{\mathtt{ntk}}(\beta) := \frac{1}{\pi}\left(\sqrt{1-\beta^2} + 2\beta(\pi - \arccos\beta)\right).$$

We show in Appendix E.2 that there exists a GPK that well-approximates $\Theta_{\mathtt{ntk}}(x,y)$ defined in (5) on datasets with bounded ℓ_2 norm. Thus, we have the following corollary of Theorem 5.2:

Corollary 5.4 (Application to NTK)**.** *For any $r > 0$ and dataset $x_1, \dots x_n \in \mathbb{R}^n$ with $\max_{i\in[n]}\|x_i\|_2^2 \le r$, any $\lambda, \epsilon > 0$, if $K \in \mathbb{R}^{n\times n}$ is the NTK kernel matrix, i.e., $K_{i,j} := \Theta_{\mathtt{ntk}}(x_i, x_j)$ as per (5), with statistical dimension $s_\lambda = \mathrm{tr}\left(K(K+\lambda I)^{-1}\right)$, then there exists an algorithm that computes $Z \in \mathbb{R}^{s\times n}$ with $s = O(\frac{s_\lambda}{\epsilon^2}\log n)$ in time $\widetilde{O}\left(\left(\frac{nr}{\epsilon\lambda}\right)^{16}\frac{s_\lambda^2 n}{\epsilon^4} + nd\right)$, such that with probability $1 - \frac{1}{\mathrm{poly}(n)}$, $Z^\top Z$ is an (ϵ,λ)-spectral approximation to K.*

Note that, for constant ϵ and any $r = (\log n)^{O(1)}$, the runtime of Corollary 5.4 is upper bounded by $\widetilde{O}\left(\left(\frac{n}{\lambda}\right)^{16}\cdot s_\lambda^2 n + nd\right)$. For comparison, the runtime of (Song et al., 2021) for spectrally approximating the NTK on datasets with unit radius $r = 1$ is $\widetilde{O}\left(n^{11/3} + nd\right)$, which means that for any $\lambda = \omega(n^{5/6})$, our runtime is strictly faster than the runtime of (Song et al., 2021). Furthermore, the random features proposed in (Zandieh et al., 2021) requires $\widetilde{O}\left((n/\lambda)\cdot nd^2\right)$ operations to spectrally approximate the NTK, which is slower than our runtime for high dimensional datasets with $d = \omega\left((n/\lambda)^{15}\right)$. Additionally, Corollary 5.4 applies to datasets with arbitrary radius r

Table 1. Approximate kernel ridge regression/classification with Gaussian and Neural Tangent kernels. We denote the ridge parameter by λ, and the number of samples or sketching dimension of different methods by s. The RMSE and classification error rates are measured on the testing sets for each task.

Data-set:	MNIST	Location of CT	
n/d	60,000 / 784	53,500 / 384	
λ / s	1 / 1,000	0.5 / 2,000	
Kernel function	$\Theta_{\texttt{ntk}}(x,y)$	$\Theta_{\texttt{ntk}}(x,y)$	$e^{-\frac{\|x-y\|^2}{40}}$
Metric	Error (%)	RMSE	RMSE
Fourier Features (Rahimi & Recht, 2008)	–	–	4.92
PolySketch (Ahle et al., 2020) (Zandieh et al., 2021)	5.92	4.87	5.05
Accelerated PolySketch (Song et al., 2021)	6.07	4.93	5.14
Adaptive Sampling (Woodruff & Zandieh, 2020)	5.87	4.72	4.76
Our Method Corollaries 5.3 and 5.4	**5.44**	**4.71**	**4.76**

while both of (Song et al., 2021) and (Zandieh et al., 2021) only apply to datasets with unit radius.

6. Experiments

In this section we apply our sampling algorithm to accelerate regression and classification on real-world datasets. We approximately solve the kernel ridge regression problem by running least squares regression on the features sampled by our algorithm. We also reduce the classification problem to regression by applying a one-hot encoding to the labels of classes and then use our fast regression method to solve it. In the experiments, we focus on ridge regression with a Gaussian kernel as well as the depth-1 Neural Tangent kernel, and compare our result from Corollaries 5.3 and 5.4 to various popular sampling and sketching methods for Gaussian and Neural Tangent kernels. The classification error rate and root mean square error (RMSE) on the testing sets are summarized in Table 1 (average over 5 trials with different random seeds). For each task, the number of features and sketching dimensions are chosen to be equal across all different methods. Thus, we can compare different methods given that the memory needed to store the approximate kernel matrices is equal for all methods.

While our theoretical results guarantee that for large enough datasets in high dimensions our method performs better than prior work, our experiments verify that even for moderately-sized datasets with dimension $d < 1000$ our method performs well. In particular, we achieve the best RMSE and classification error rate compared to all other methods under the condition that the number of sampled features or sketching dimension is fixed for each method. We remark

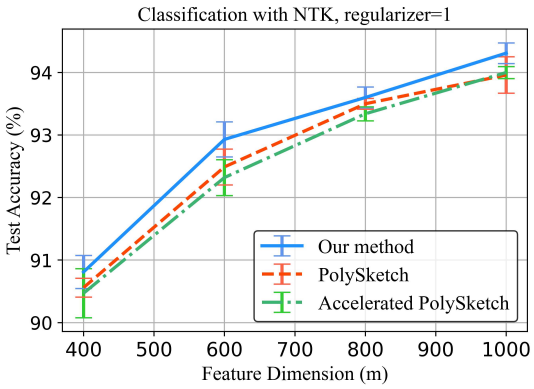

Figure 1. Approximate classification of the MNIST dataset using depth-1 Neural Tangent KRR. The ridge parameter is $\lambda = 1$. The classification error rates are measured on the testing set.

that the Fourier features method (Rahimi & Recht, 2008) only applies to shift invariant kernels such as the Gaussian kernel and cannot be used for Neural Tangent kernels. On the other hand, the sketching methods of (Ahle et al., 2020) and (Song et al., 2021) can be used to sketch the Taylor expansion of the NTK, as was previously done in (Zandieh et al., 2021).

Accuracy/memory trade-off. Figure 1 shows the trade-off of various methods for MNIST classification using the NTK kernel function. We plot the testing set accuracy as a function of the number of samples or sketching dimension, which is a parameter that directly controls the memory usage of different methods. It can been seen that our method has the best accuracy/memory trade-off.

Acknowledgements

David Woodruff would like to thank NSF grant No. CCF-1815840, NIH grant 5401 HG 10798-2, ONR grant N00014-18-1-2562, and a Simons Investigator Award. Amir Zandieh was supported by the Swiss NSF grant No. P2ELP2_195140.

References

Ahle, T. D., Kapralov, M., Knudsen, J. B., Pagh, R., Velingker, A., Woodruff, D. P., and Zandieh, A. Oblivious sketching of high-degree polynomial kernels. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 141–160. SIAM, 2020.

Ailon, N. and Chazelle, B. The fast johnson–lindenstrauss transform and approximate nearest neighbors. *SIAM Journal on computing*, 39(1):302–322, 2009.

Avron, H., Nguyen, H., and Woodruff, D. Subspace em-

beddings for the polynomial kernel. *Advances in neural information processing systems*, 27, 2014.

Avron, H., Kapralov, M., Musco, C., Musco, C., Velingker, A., and Zandieh, A. Random fourier features for kernel ridge regression: Approximation bounds and statistical guarantees. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 253–262. JMLR. org, 2017.

Avron, H., Kapralov, M., Musco, C., Musco, C., Velingker, A., and Zandieh, A. A universal sampling method for reconstructing signals with simple fourier transforms. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1051–1063, 2019.

Charikar, M., Chen, K., and Farach-Colton, M. Finding frequent items in data streams. In *International Colloquium on Automata, Languages, and Programming*, pp. 693–703. Springer, 2002.

Cohen, M. B., Lee, Y. T., Musco, C., Musco, C., Peng, R., and Sidford, A. Uniform sampling for matrix approximation. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, pp. 181–190, 2015.

Dasgupta, S. and Gupta, A. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

El Alaoui, A. and Mahoney, M. W. Fast randomized kernel methods with statistical guarantees. *stat*, 1050:2, 2014.

Haagerup, U. and Musat, M. On the best constants in noncommutative khintchine-type inequalities. *Journal of Functional Analysis*, 250(2):588–624, 2007.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Li, M., Miller, G. L., and Peng, R. Iterative row sampling. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pp. 127–136. IEEE, 2013.

Meister, M., Sarlos, T., and Woodruff, D. Tight dimensionality reduction for sketching low degree polynomial kernels. *Advances in Neural Information Processing Systems*, 32: 9475–9486, 2019.

Musco, C. and Musco, C. Recursive sampling for the nystrom method. In *Advances in Neural Information Processing Systems*, pp. 3833–3845, 2017.

Pham, N. and Pagh, R. Fast and scalable polynomial kernels via explicit feature maps. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 239–247, 2013.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pp. 1177–1184, 2008.

Rahimi, A. and Recht, B. Random Features for Large-Scale Kernel Machines. 2009.

Song, Z., Woodruff, D., Yu, Z., and Zhang, L. Fast sketching of polynomial kernels of polynomial degree. In *International Conference on Machine Learning*, pp. 9812–9823. PMLR, 2021.

Williams, C. and Seeger, M. Using the nystroem method to speed up kernel machines. *Advances in Neural Information Processing Systems 13*, 2001.

Woodruff, D. and Zandieh, A. Near input sparsity time kernel embeddings via adaptive sampling. In *International Conference on Machine Learning*, pp. 10324–10333. PMLR, 2020.

Zandieh, A., Han, I., Avron, H., Shoham, N., Kim, C., and Shin, J. Scaling neural tangent kernels via sketching and random features. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=vIRFiA658rh.

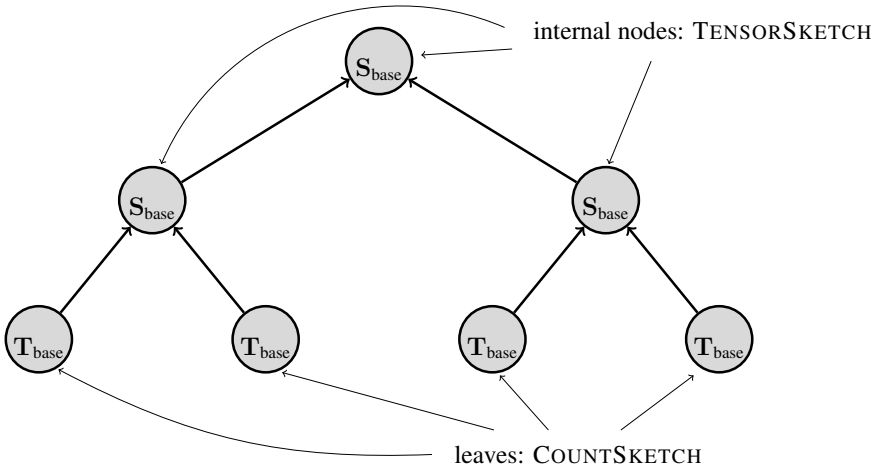

Figure 2. The structure of sketch S^q proposed in Theorem 1.1 of (Ahle et al., 2020): the sketch matrices in nodes of the tree labeled with S_{base} and T_{base} are independent instances of degree-2 TENSORSKETCH and COUNTSKETCH, respectively.

A. Preliminary Sketching Results

In this section we provide preliminary sketching results. In particular, we provide a proof of Lemma 2.3.

Proof of Lemma 2.3: By invoking Corollary 4.1 of (Ahle et al., 2020), we find that there exists a random sketch $S^q \in \mathbb{R}^{m \times d^q}$ such that if $m = C \cdot q \cdot \varepsilon^{-2}$ for some absolute constant C, then this sketch satisfies the $(\epsilon, 1/20, 2)$-JL-moment property. It follows from the definition of the JL-moment property along with Minkowski's Inequality that for any $Y \in \mathbb{R}^{d^q \times n}$,

$$\mathbb{E}\left[\left|\|S^qY\|_F^2 - \|Y\|_F^2\right|^2\right] \leq \epsilon^2/20 \cdot \|Y\|_F^4 .$$

Thus, by applying Markov's inequality on $\left|\|S^qY\|_F^2 - \|Y\|_F^2\right|^2$, we find that

$$\Pr\left[\|S^qY\|_F^2 \in (1 \pm \varepsilon)\|Y\|_F^2\right] \geq 19/20.$$

This immediately proves the first statement of the lemma.

It was shown in (Ahle et al., 2020) that the sketch S^q can be represented by a binary tree with q leaves. As shown in Figure 2, the leaves are independent copies of COUNTSKETCH and the internal nodes are independent instances of degree-2 TENSORSKETCH (Pham & Pagh, 2013), which can sketch 2-fold tensor products efficiently. The sketch S^q can be applied to tensor product vectors of the form $u_1 \otimes u_2 \otimes \dots u_q$ by recursive application of $O(q)$ independent instances of COUUNTSKETCH (Charikar et al., 2002) and degree-2 TENSORSKETCH (Pham & Pagh, 2013) on vectors u_i and their sketched versions. The use of COUNTSKETCH in the leaves of this sketch structure ensures input sparsity runtime for sketching sparse input vectors.

Runtime analysis: By Theorem 1.1 of (Ahle et al., 2020), for any collection of vectors $u_1, u_2, \dots u_q \in \mathbb{R}^d$, $S^q\left(u_1 \otimes u_2 \otimes \dots u_q\right)$ can be computed in time $O\left(qm \log m + \sum_{j=1}^q \text{nnz}(u_j)\right)$. From the binary tree structure of the sketch, shown in Figure 2, it follows that once we compute $S^q\left(u_1 \otimes u_2 \otimes \dots u_q\right)$, then $S^q\left(e_1 \otimes u_2 \otimes u_3 \otimes \dots u_q\right)$ can be computed by updating the path from one of the leaves to the root of the binary tree. This exactly amounts to applying an instance of COUNTSKETCH on e_1 and then applying $O(\log q)$ instances of degree-2 TENSORSKETCH on the intermediate nodes of the tree. This can be computed in a total additional runtime of $O(m \log m \log q)$. By this argument, it follows that $S^q\left(e_1^{\otimes j} \otimes u_{j+1} \otimes u_{j+2} \dots u_q\right)$ can be computed sequentially for all $j = 0, 1, 2, \cdots q$ in total time $O\left(qm \log m \log q + \sum_{j=1}^q \text{nnz}(u_j)\right)$. By plugging in the value $m = O\left(\frac{q}{\varepsilon^2}\right)$, this runtime will be upper bounded by $O\left(\frac{q^2 \log^2 \frac{q}{\varepsilon}}{\varepsilon^2} + \sum_{j=1}^q \text{nnz}(u_j)\right)$, which gives the second statement of the lemma.

□

In order to prove our main result about SRHT with shared random signs in Lemma 4.1, we use Khintchine's inequality. We provide a formal statement of this inequality in the following lemma.

Lemma A.1 (Khintchine's inequality (Haagerup & Musat, 2007)). *Let t be a positive integer, $x \in \mathbb{R}^d$, and $(\sigma_i)_{i\in[d]}$ be independent Rademacher ± 1 random variables. Then*

$$\left(E\left[|\langle \sigma, x\rangle|^t\right]\right)^{1/t} \leq C_t \|x\|_2,$$

where $C_t \leq \sqrt{2}\left(\frac{\Gamma((t+1)/2)}{\sqrt{\pi}}\right)^{1/t} \leq \sqrt{t}$ for all $t \geq 1$. Consequently, by Minkowski's Inequality along with Markov's inequality, for any $\delta > 0$ and any matrix $X \in \mathbb{R}^{d\times n}$, we have

$$\Pr\left[\left\|X^\top \cdot \sigma\right\|_2 \geq 2\sqrt{\log_2 \frac{1}{\delta}} \cdot \|X\|_F\right] \leq \delta.$$

A.1. Proof of Lemma 3.1

Let $P_{i,j}$ be the matrices defined in line 3 of Algorithm 2. For every $V \in \mathbb{R}^{n\times r}$, we can write,

$$P_{i,j} \cdot V = Q_i \cdot S_i^q \cdot \left(\left(E_1^{\otimes j} \otimes X^{(j+1)} \otimes X^{(j+2)} \otimes \dots X^{(q)}\right) \cdot V\right),$$

where S_i^q is an instance of degree-q PolySketch and Q_i is an SRHT . By Lemma 2.3 and Lemma 2.4 and a union bound, for every fixed $i \in [T]$ and $j \in \{0, 1, 2, \dots q\}$ the following holds,

$$\Pr\left[\|P_{i,j} \cdot V\|_F^2 \in (1 \pm \epsilon)\left\|\left(E_1^{\otimes j} \otimes X^{(j+1)} \otimes X^{(j+2)} \otimes \dots X^{(q)}\right) \cdot V\right\|_F^2\right] \geq 9/10 \tag{6}$$

Using the properties of tensor products and the definition of matrix E_1 we have,

$$\left\|\left(E_1^{\otimes j} \otimes X^{(j+1)} \otimes X^{(j+2)} \otimes \dots X^{(q)}\right) \cdot V\right\|_F^2 = \left\|\left(X^{(j+1)} \otimes X^{(j+2)} \otimes \dots X^{(q)}\right) \cdot V\right\|_F^2$$

Because $\tilde{z}_j$ is defined as the median over $T = \Omega(\log n)$ independent copies in line 4 of Algorithm 2, using the above equality and (6) we have,

$$\Pr\left[\tilde{z} \in (1 \pm \epsilon)\left\|\left(X^{(j+1)} \otimes X^{(j+2)} \otimes \dots X^{(q)}\right) \cdot V\right\|_F^2\right] \geq 1 - \frac{1}{\mathrm{poly}(n)}.$$

This proves the first statement of the lemma.

Runtime and Memory: The time to compute $P_{i,j}$ for a fixed i and all $j = 0, 1, \dots q$ is $O\left(\frac{q^2 \log^2 \frac{q}{\epsilon}}{\epsilon^2} \cdot n + \sum_{j=1}^q \mathrm{nnz}\left(X^{(j)}\right)\right)$, by Lemma 2.3 and Lemma 2.4. Therefore, the total time to compute $P_{i,j}$ for all $i \in [T]$ and all $j = 0, 1, \dots q$ is $O\left(\frac{q^2 \log^2 \frac{q}{\epsilon}}{\epsilon^2} \cdot n \log n + \log n \cdot \sum_{j=1}^q \mathrm{nnz}\left(X^{(j)}\right)\right)$. Since matrices $P_{i,j}$ are of size $m' \times n$, the total memory needed to store them for all i and j is $O\left(\frac{q \log(1/\epsilon)}{\epsilon^2} \cdot n \log n\right)$. Finally note that the runtime of $\textsc{Query}(V, j)$ is dominated by time needed to compute the product $P_{i,j} \cdot V$ for $i \in [T]$. This can be done in $O\left(\frac{\log(1/\epsilon)}{\epsilon^2} \cdot \log n \cdot \mathrm{nnz}(V)\right)$ operations.

B. Spectral Approximation to Tensor Product Matrices $\Phi = \bigotimes_{j=1}^q X^{(j)}$

In this section we design the RowSampler procedure which can perform *row norm sampling* as per Definition 2.5 on $\Phi(B^\top B + \lambda I)^{-1/2}$ for $\Phi = \bigotimes_{j=1}^q X^{(j)}$ using $\widetilde{O}\left(\sum_i \mathrm{nnz}\left(X^{(i)}\right)\right)$ runtime. Our primitive crucially relies on TensorNorMDS, given in Algorithm 2, to quickly estimate norm queries of the form $\left\|\left(\bigotimes_{j=1}^q X^{(j)}\right) V\right\|_F^2$.

Algorithm 4 ROWSAMPLER for $\Phi = \bigotimes_{j=1}^{q} X^{(j)}$

input: $q, s \in \mathbb{Z}_+$, $X^{(1)}, \dots X^{(q)} \in \mathbb{R}^{d \times n}$, $B \in \mathbb{R}^{m \times n}$, $\lambda > 0$
output: Sampling matrix $S \in \mathbb{R}^{s \times d^q}$

1: Generate $H \in \mathbb{R}^{d' \times n}$ with i.i.d. normal entries with $d' = C_1 q \log n$ rows
2: $M \leftarrow H \cdot (B^\top B + \lambda I)^{-1/2}$
3: Let TNORM be the DS in Algorithm 2 for inputs $\left(X^{(1)}, X^{(2)}, \dots, X^{(q)}, M\right)$ and $\epsilon = \frac{1}{20q}$
4: Let $h : [d] \to [s']$ be a fully independent and uniform hash function with $s' = \lceil q^2 s \rceil$ buckets
5: Define the set $h^{-1}(r) := \{j \in [d] : h(j) = r\}$ for every $r \in [s']$
6: For every $r \in [s']$ and $k \in [m']$, let $G_r^k \in \mathbb{R}^{n' \times d_r}$ be independent instances of degree-1 POLYSKETCH as per Lemma 2.3, where $d_r = |h^{-1}(r)|$, $n' = C_2 q^2$, and $m' = C_3 \log n$
7: $W_{r,k}^a \leftarrow G_r^k \cdot X^{(a)}_{h^{-1}(r),\star}$ for every $a \in [q]$, $k \in [m']$, and $r \in [s']$
8: **for** $\ell = 1$ to s **do**
9: $\quad D^1 \leftarrow I_n$ and $\beta_\ell \leftarrow s$
10: $\quad$ **for** $a = 1$ to q **do**
11: $\quad\quad L_{r,k}^a \leftarrow D^a \cdot W_{r,k}^{a\top}$ for every $k \in [m']$, and $r \in [s']$
12: $\quad\quad p_r^a \leftarrow \text{MEDIAN}_{k \in [m']} \left\{ \text{TNORM.QUERY}(L_{r,k}^a, a) \right\}$ for every $r \in [s']$
13: $\quad\quad p_r^a \leftarrow p_r^a / \sum_{t=1}^{s'} p_t^a$ for every $r \in [s']$
14: $\quad\quad$ Sample $t \in [s']$ from distribution $\{p_r^a\}_{r=1}^{s'}$
15: $\quad\quad q_i^a \leftarrow \text{TNORM.QUERY}\left(D^a \cdot X_{i,\star}^{(a)\top}, a\right)$ for every $i \in h^{-1}(t)$
16: $\quad\quad q_i^a \leftarrow q_i^a / \sum_{j \in h^{-1}(t)} q_j^a$ for every $i \in h^{-1}(t)$
17: $\quad\quad$ Sample $i_a \in [d]$ from distribution $\{q_i^a\}_{i \in h^{-1}(t)}$
18: $\quad\quad D^{a+1} \leftarrow D^a \cdot \text{diag}\left(X_{i_a,\star}^{(a)}\right)$
19: $\quad\quad \beta_\ell \leftarrow \beta_\ell \cdot p_t^a q_{i_a}^{a,t}$
20: $\quad$ Let the ℓ^{th} row of S be $\beta_\ell^{-1/2} \left(e_{i_1} \otimes e_{i_2} \otimes \cdots e_{i_q}\right)^\top$
21: **return** S

Overview of Algorithm 4: The goal of ROWSAMPLER is to generate a sample $(i_1, i_2, \cdots i_q) \in [d]^q$ with probability proportional to the squared norm of the row $(i_1, \cdots i_q)$ of matrix $\left(\bigotimes_{j=1}^{q} X^{(j)}\right) \cdot (B^\top B + \lambda I)^{-1/2}$. Because $(B^\top B + \lambda I)^{-1/2}$ has a large $n \times n$ size, we first compress it using random projection techniques without perturbing the row norm distribution of $\left(\bigotimes_{j=1}^{q} X^{(j)}\right) \cdot (B^\top B + \lambda I)^{-1/2}$ too much. This can be done by applying a JL-transformation to the rows of this matrix (see, e.g., (Dasgupta & Gupta, 2003)). Let $H \in \mathbb{R}^{d' \times n}$ be a random matrix with i.i.d. normal entries with $d' = C_1 q \log_2 n$ rows. With probability $1 - \frac{1}{\text{poly}(n^q)}$ the norm of each row of the sketched matrix $\left(\bigotimes_{j=1}^{q} X^{(j)}\right) \cdot (B^\top B + \lambda I)^{-1/2} \cdot H^\top$ is preserved up to a (1 ± 0.1) factor and hence by a union bound, with probability $1 - \frac{1}{\text{poly}(n^q)}$, all row norms of the sketched matrix are within a (1 ± 0.1) factor of the original row norms. This is done in line 2 of the algorithm by computing $M := H \cdot (B^\top B + \lambda I)^{-1/2}$, which can be computed quickly since matrices B and H have few rows.

Now the problem is reduced to performing row norm sampling on $\left(\bigotimes_{j=1}^{q} X^{(j)}\right) \cdot M^\top$. Note that computing the exact row norms of this matrix is out of the question since it has a huge d^q number of rows. However, by using TENSORNORMDS that we designed in Algorithm 2 and exploiting the properties of tensor products we can approximately generate samples from the row norm distribution in near input sparsity time as follows:

First note that by basic properties of tensor products, the entries of $\left(X^{(1)} \otimes X^{(2)} \dots X^{(q)}\right) \cdot M^\top$ are in bijective correspondence with the entries of $\left(X^{(1)} \otimes M\right) \cdot \left(X^{(2)} \otimes X^{(3)} \dots X^{(q)}\right)^\top$. More precisely, the entry at row $(i_1, i_2, \cdots i_q)$ and column j of $\left(X^{(1)} \otimes X^{(2)} \dots X^{(q)}\right) \cdot M^\top$ is equal to the entry at row (i_1, j) and column $(i_2, \dots i_q)$ of $\left(X^{(1)} \otimes M\right) \cdot \left(X^{(2)} \otimes \dots X^{(q)}\right)^\top$.

Therefore, it is enough to have a procedure to sample $(i_1, i_2, \dots i_q)$ with probability proportional to the squared norm of

column $(i_2, \dots i_q)$ of matrix $\left(M \cdot \operatorname{diag}\left(X^{(1)}_{i_1,\star}\right)\right) \cdot \left(X^{(2)} \otimes \dots X^{(q)}\right)^\top$ for every $i_1 \in [d]$. To this end, we first sample an index i_1 with probability proportional to the squared Frobenius norm of $\left(M \cdot \operatorname{diag}\left(X^{(1)}_{i_1,\star}\right)\right) \cdot \left(X^{(2)} \otimes \dots X^{(q)}\right)^\top$, and then perform column norm sampling on the sampled matrix. We can cheaply estimate the Frobenius norms of matrices $\left(M \cdot \operatorname{diag}\left(X^{(1)}_{i_1,\star}\right)\right) \cdot \left(X^{(2)} \otimes \dots X^{(q)}\right)^\top$ up to $\left(1 \pm \frac{1}{20q}\right)$ perturbation using TENSORNORMDS (Algorithm 2).

After this point, we will have an index $i_1 \in [d]$ sampled from the right distribution and all that is left to do is to carry out row norm sampling on $\left(X^{(2)} \otimes \dots X^{(q)}\right) \cdot \left(M \cdot \operatorname{diag}\left(X^{(1)}_{i_1,\star}\right)\right)^\top$. Note that we have made progress because this matrix has d^{q-1} rows, meaning that we have reduced the size of our problem by a factor of d. Algorithm 4 recursively repeats this process of reshaping, norm estimation, and sampling q times until having all q indices $i_1, i_2, \cdots i_q$.

Note that the actual procedure requires more work because we need to generate s i.i.d. samples with the row norm distribution and to ensure that the runtime does not lose a multiplicative factor of s, resulting in $s \cdot \sum_{j \in [q]} \operatorname{nnz}\left(X^{(j)}\right)$ total time, we need to do extra sketching and a random partitioning of the rows of the datasets to $q^2 s$ buckets. Moreover, we use the median trick to boost the success probabilities of our randomized operations, when needed.

The formal guarantee on Algorithm 4 is given in the following lemma.

Lemma B.1. *For any matrices $X^{(1)}, X^{(2)}, \dots X^{(q)} \in \mathbb{R}^{d \times n}$ and $B \in \mathbb{R}^{m \times n}$, any $\lambda > 0$ and any positive integers q, s, with probability at least $1 - \frac{1}{\operatorname{poly}(n)}$, Algorithm 4 outputs a ranks-s row norm sampler for the matrix $\left(X^{(1)} \otimes X^{(2)} \otimes \dots X^{(q)}\right) \cdot (B^\top B + \lambda I)^{-1/2}$ as per Definition 2.5 in time $O\left(m^2 n + q^7 s^2 n \log^3 n + \log^3 n \log q \sum_{j=1}^{q} \operatorname{nnz}\left(X^{(j)}\right)\right)$.*

Proof. All rows of the sampling matrix $S \in \mathbb{R}^{s \times d^q}$ (the output of Algorithm 4) have independent and identical distributions because for each $\ell \in [s]$, the ℓ^{th} row of the matrix S is constructed by sampling indices $i_1, i_2, \cdots i_q$ in line 17 completely independent of the sampled values for other rows $\ell' \neq \ell$. Thus, it is enough to consider the distribution of the ℓ^{th} row of S for some arbitrary $\ell \in [s]$.

Let $I := (I_1, I_2, \cdots I_q)$ be a vector-valued random variable that takes values in $[d]^q$ with the following conditional probability distribution for every $a = 1, 2, \cdots q$ and every $i \in [d]$,

$$\Pr\left[I_a = i | I_1 = i_1, I_2 = i_2, \cdots I_{a-1} = i_{a-1}\right] := p^a_{h(i)} \cdot q^a_i, \tag{7}$$

where distributions $\{p^a_r\}_{r \in [s']}$ and $\{q^a_i\}_{i \in h^{-1}(t)}$ for every $t \in [s']$ are defined as per lines 13 and 16 of the algorithm. One can see that the random vector $(i_1, i_2, \cdots i_q)$ obtained by stitching together the random indices sampled in line 17 of the algorithm, is in fact a copy of the random variable I defined above.

Let β_ℓ be the quantity computed in line 19 of the algorithm. If $i_1, i_2, \cdots i_q \in [d]$ are the indices sampled in line 17 of the algorithm, then using the conditional distribution of I in (7), we find that the value of β_ℓ is equal to the following,

$$\begin{aligned}
\beta_\ell &= s \cdot \prod_{a=1}^{q} p^a_{h(i_a)} q^a_{i_a} \\
&= s \cdot \prod_{a=1}^{q} \Pr\left[I_a = i_a | I_1 = i_1, I_2 = i_2, \cdots I_{a-1} = i_{a-1}\right] \\
&= s \cdot \Pr\left[I = (i_1, i_2, \dots i_q)\right],
\end{aligned}$$

where p^a and q^a are the distributions computed in lines 13 and 16 of the algorithm. Hence, for any $i_1, i_2, \cdots i_q \in [d]$, the distribution of $S_{\ell,\star}$ is,

$$\begin{aligned}
&\Pr\left[S_{\ell,\star} = \beta_\ell^{-1/2} (e_{i_1} \otimes e_{i_2} \otimes \cdots e_{i_q})^\top\right] \\
&\qquad = \Pr\left[I = (i_1, i_2, \dots i_q)\right] = \frac{\beta_\ell}{s}.
\end{aligned} \tag{8}$$

We will use (8) later.

By Lemma 3.1 and the way TNORM is constructed in line 3 of the algorithm, we have the following inequalities for any $r \in [s']$, $k \in [m']$, $i \in [d]$, and any $a = 1, 2, \dots q$, with probability at least $1 - \frac{1}{\text{poly}(n)}$,

$$\text{TNORM.QUERY}\left(L^a_{r,k}, a\right) \in \left(1 \pm \frac{1}{20q}\right) \left\| \left(X^{(a+1)} \otimes \dots X^{(q)} \otimes M\right) D^a W^{a\top}_{r,k} \right\|_F^2, \tag{9}$$

$$\text{TNORM.QUERY}\left(D^a \cdot X^{(a)\top}_{i,\star}, a\right) \in \left(1 \pm \frac{1}{20q}\right) \cdot \left\| \left(X^{(a+1)} \otimes \dots X^{(q)} \otimes M\right) D^a \cdot X^{(a)\top}_{i,\star} \right\|_2^2. \tag{10}$$

By union bounding over $qds'm'$ events, with probability at least $1 - \frac{1}{\text{poly}(n)}$, (9) and (10) hold simultaneously for all $a \in [q]$, $k \in [m']$, $i \in [d]$, and all $r \in [s']$.

Furthermore, note that $W^a_{r,k}$ is defined in line 7 as $W^a_{r,k} = G^k_r \cdot X^{(a)}_{h^{-1}(r),\star}$, where G^k_r is an instance of the degree-1 POLYSKETCH as per Lemma 2.3 with target dimension $n' = C_2 q^2$. By the first statement of Lemma 2.3, the POLYSKETCH G^k_r approximately preserves the Frobenius norm of any fixed matrix with constant probability. In particular, for every $a \in [q], r \in [s'], k \in [m']$, with probability at least $9/10$ the following holds,

$$\left\| \left(X^{(a+1)} \otimes \dots X^{(q)} \otimes M\right) D^a W^{a\top}_{r,k} \right\|_F^2 \in \left(1 \pm \frac{1}{50q}\right) \left\| \left(X^{(a+1)} \otimes \dots X^{(q)} \otimes M\right) D^a \left(X^{(a)}_{h^{-1}(r),\star}\right)^\top \right\|_F^2. \tag{11}$$

By taking the median of $m' = \Omega(\log n)$ independent instances of G^k_r, the success probability in (11) gets boosted. Thus, by combining this inequality with (9) using a union bound, and applying the median trick, with probability at least $1 - \frac{1}{\text{poly}(n)}$ the following holds simultaneously for all $a \in [q]$ and $r \in [s']$,

$$\text{MEDIAN}_{k \in [m']} \left\{ \text{TNORM.QUERY}\left(L^a_{r,k}, a\right) \right\} \in \left(1 \pm \frac{1}{14q}\right) \left\| \left(X^{(a+1)} \otimes \dots X^{(q)}\right) D^a \left(X^{(a)}_{h^{-1}(r),\star} \otimes M\right)^\top \right\|_F^2 \tag{12}$$

Note that to obtain the above inequality we used the property of tensor products regarding the bijective correspondence between entries of $\left(X^{(a+1)} \otimes \dots X^{(q)} \otimes M\right) D^a \left(X^{(a)}_{h^{-1}(r),\star}\right)^\top$ and $\left(X^{(a+1)} \otimes \dots X^{(q)}\right) D^a \left(X^{(a)}_{h^{-1}(r),\star} \otimes M\right)^\top$

By plugging the above inequality along with (10) into (7), we conclude that with high probability the following bound holds simultaneously for all $a \in [q]$,

$$\begin{aligned} &\Pr[I_a = i | I_1 = i_1, I_2 = i_2, \cdots I_{a-1} = i_{a-1}] \\ &\quad \geq \left(1 - \frac{1}{5q}\right) \cdot \frac{\left\| \left(X^{(a+1)} \otimes \dots X^{(q)}\right) D^a \text{diag}\left(X^{(a)}_{i,\star}\right) M^\top \right\|_F^2}{\left\| \left(X^{(a+1)} \otimes \dots X^{(q)}\right) D^a \left(X^{(a)} \otimes M\right)^\top \right\|_F^2}. \end{aligned} \tag{13}$$

Again to obtain the above inequality we used the property of tensor products regarding the bijective correspondence between the entries of vector $\left(X^{(a+1)} \otimes \dots X^{(q)} \otimes M\right) D^a \cdot X^{(a)\top}_{i,\star}$ and matrix $\left(X^{(a+1)} \otimes \dots X^{(q)}\right) D^a \text{diag}\left(X^{(a)}_{i,\star}\right) M^\top$

It follows from the properties of tensor products and the definition of D^a in line 18 of the algorithm, that

$$\begin{aligned} \left\| \left(X^{(a+2)} \otimes \dots X^{(q)}\right) D^{a+1} \left(X^{(a+1)} \otimes M\right)^\top \right\|_F^2 &= \left\| \left(X^{(a+2)} \otimes \dots X^{(q)}\right) D^a \text{diag}\left(X^{(a)}_{i_a,\star}\right) \left(X^{(a+1)} \otimes M\right)^\top \right\|_F^2 \\ &= \left\| \left(X^{(a+1)} \otimes X^{(a+2)} \otimes \dots X^{(q)}\right) D^a \text{diag}\left(X^{(a)}_{i_a,\star}\right) M^\top \right\|_2^2 \end{aligned}$$

Using this equality and inequality (13), we have:

$$\begin{aligned}
\Pr\left[I=(i_1,i_2,\cdots i_q)\right] &= \prod_{a=1}^{q}\Pr\left[I_a=i_a|I_1=i_1,\cdots I_{a-1}=i_{a-1}\right] \\
&\geq \prod_{a=1}^{q}\left(1-\frac{1}{5q}\right)\frac{\left\|\left(X^{(a+1)}\otimes\ldots X^{(q)}\right)D^a\cdot\operatorname{diag}\left(X^{(a)}_{i_a,\star}\right)M^\top\right\|_F^2}{\left\|\left(X^{(a+1)}\otimes\ldots X^{(q)}\right)D^a\left(X^{(a)}\otimes M\right)^\top\right\|_F^2} \\
&\geq \frac{3}{4}\cdot\frac{\left\|\mathbf{1}_n^\top\cdot D^q\cdot\operatorname{diag}\left(X^{(q)}_{i_q,\star}\right)M^\top\right\|_F^2}{\left\|\left(X^{(2)}\otimes\ldots X^{(q)}\right)D^1\left(X^{(1)}\otimes M\right)^\top\right\|_F^2} \\
&= \frac{3}{4}\cdot\frac{\left\|\left[\left(X^{(1)}\otimes X^{(2)}\otimes\ldots X^{(q)}\right)\cdot M^\top\right]_{(i_1,i_2,\cdots i_q),\star}\right\|_2^2}{\left\|\left(X^{(1)}\otimes X^{(2)}\otimes\ldots X^{(q)}\right)\cdot M^\top\right\|_F^2}
\end{aligned} \tag{14}$$

By plugging (14) back in (8) we find that,

$$\Pr\left[S_{\ell,\star}=\beta_\ell^{-1/2}(e_{i_1}\otimes e_{i_2}\otimes\cdots e_{i_q})^\top\right]\geq\frac{3}{4}\cdot\frac{\left\|\left[\left(X^{(1)}\otimes\ldots X^{(q)}\right)\cdot M^\top\right]_{(i_1,i_2,\cdots i_q),\star}\right\|^2}{\left\|\left(X^{(1)}\otimes\ldots X^{(q)}\right)\cdot M^\top\right\|_F^2}$$

Matrix M is defined as $M=H\cdot(B^\top B+\lambda I)^{-1/2}$ where H is a random matrix with i.i.d. Gaussian entries with $d'=C_1q\log n$ rows. Therefore, H is a JL-transform, so for every $(i_1,i_2,\cdots i_q)\in[d]^q$, with probability $1-\frac{1}{\text{poly}(n^q)}$,

$$(d')^{-1}\left\|\left[\left(X^{(1)}\otimes\ldots X^{(q)}\right)\cdot M^\top\right]_{(i_1,i_2,\cdots i_q),\star}\right\|_2^2\in(1\pm0.1)\left\|\left[X^{(1)}\otimes\ldots X^{(q)}\right]_{(i_1,i_2,\cdots i_q),\star}(B^\top B+\lambda I)^{-1/2}\right\|_2^2.$$

Therefore, by union bounding over d^q rows of $\left(X^{(1)}\otimes\ldots X^{(q)}\right)\cdot M^\top$, the above holds simultaneously for all $(i_1,i_2,\cdots i_q)\in[d]^q$ with probability $1-\frac{1}{\text{poly}(n^q)}$. Therefore, with high probability in n,

$$\begin{aligned}
&\Pr\left[S_{\ell,\star}=\beta_\ell^{-1/2}(e_{i_1}\otimes e_{i_2}\otimes\cdots e_{i_q})^\top\right] \\
&\geq\frac{1}{2}\cdot\frac{\left\|\left[\left(X^{(1)}\otimes\ldots X^{(q)}\right)\cdot(B^\top B+\lambda I)^{-1/2}\right]_{(i_1,i_2,\cdots i_q),\star}\right\|_2^2}{\left\|\left(X^{(1)}\otimes\ldots X^{(q)}\right)\cdot(B^\top B+\lambda I)^{-1/2}\right\|_F^2}
\end{aligned}$$

Because $\frac{\beta_\ell}{s}$ is the probability of sampling row $(i_1,i_2,\cdots i_q)$ of $\left(X^{(1)}\otimes\ldots X^{(q)}\right)\cdot(B^\top B+\lambda I)^{-1/2}$, the above inequality proves that with high probability, matrix S is a rank-s row norm sampler for $\left(X^{(1)}\otimes\ldots X^{(q)}\right)\cdot(B^\top B+\lambda I)^{-1/2}$ as in Definition 2.5.

Runtime: One of the expensive steps of this algorithm is the computation of M in line 2 which takes $O(m^2n+qmn\log n)$ operations since B has rank at most m. Another expensive step is the computation of the TNORM data-structure in line 3. By Lemma 3.1, this DS for $\epsilon=\frac{1}{20q}$ can be formed in time $O\left(q^4\log^2 q\cdot n\log n+\log n\cdot\sum_{j=1}^q\text{nnz}\left(X^{(j)}\right)\right)$.

By Lemma 2.3, matrices $W^a_{r,k}$ for all $r\in[s']$, $k\in[m']$ and $a\in[q]$ in line 7 of the algorithm can be computed in total time $O\left(q^3s'n\log^2 n+\log n\cdot\sum_{j=1}^q\text{nnz}\left(X^{(j)}\right)\right)$.

The matrix $W^a_{r,k}$ for every $k\in[m']$, and $r\in[s']$, has size $O(q^2)\times n$. Thus, by Lemma 3.1, computing the distribution $\{p^a_r\}_{r=1}^{s'}$ in line 13 takes time $O\left(q^4s'\cdot n\log^2 n\log q\right)$ for a fixed $a\in[q]$ and a fixed $\ell\in[s]$. Therefore, the total time to compute this distribution for all a and ℓ is $O\left(q^7s^2\cdot n\log^2 n\log q\right)$.

The runtime of computing the distribution $\{q^a_i\}_{i\in h^{-1}(t)}$ in line 16 depends on the sparsity of $X^{(a)}_{h^{-1}(t),\star}$, i.e., nnz $\left(X^{(a)}_{h^{-1}(t),\star}\right)$. To bound the sparsity of $X^{(a)}_{h^{-1}(t),\star}$, note that, nnz $\left(X^{(a)}_{h^{-1}(t),\star}\right)=\sum_{i=1}^d\mathbb{1}_{\{i\in h^{-1}(t)\}}\cdot\text{nnz}\left(X^{(a)}_{i,\star}\right)$. Since the hash function

h is fully independent, by invoking Bernstein's inequality, we find that for every $t \in [s']$ and $a \in [q]$, with high probability in n, $\operatorname{nnz}\left(X^{(a)}_{h^{-1}(t),\star}\right) = O\left(\left(\operatorname{nnz}\left(X^{(a)}\right)/s' + n\right)\log n\right)$. By union bounding over qs' events, with high probability in n, $\operatorname{nnz}\left(X^{(a)}_{h^{-1}(t),\star}\right) = O\left(\left(\operatorname{nnz}\left(X^{(a)}\right)/s' + n\right)\log n\right)$, simultaneously for all $t \in [s']$ and $a \in [q]$.

Therefore, by Lemma 3.1, the distribution $\{q_i^a\}_{i\in h^{-1}(t)}$ in line 16 of the algorithm can be computed in total time $O\left(q^3 sn \log^3 n \log q + \log^3 n \log q \cdot \sum_{j=1}^q \operatorname{nnz}\left(X^{(j)}\right)\right)$ for all $a \in [q]$ and all $\ell \in [s]$.

The total runtime of Algorithm 4 is thus $O\left(m^2 n + q^7 s^2 n \log^2 n \log q + \log^3 n \log q \cdot \sum_{j=1}^q \operatorname{nnz}\left(X^{(j)}\right)\right)$. □

Now we can prove our main theorem about spectrally approximating the Gram matrix $\Phi^\top\Phi$ for matrices of the form $\Phi = \bigotimes_{j=1}^q X^{(j)}$ using nearly $\sum_i \operatorname{nnz}\left(X^{(i)}\right)$ runtime.

Proof of Theorem 2.7: The theorem follows by invoking Lemmas 2.6 and B.1. To find the sampling matrix Π, run Algorithm 1 on Φ with $\mu = s_\lambda$ and for the RowSampler primitive, invoke Algorithm 4. By Lemma B.1, Algorithm 4 outputs a row norm sampler as per Definition 2.5 with probability $1 - \frac{1}{\operatorname{poly}(n)}$. Therefore, since the total number of times Algorithm 4 is invoked by Algorithm 1 is $\log \frac{\|\Phi\|_F^2}{\epsilon\lambda} = O(\log n)$, by a union bound, the preconditions of Lemma 2.6 are satisfied with high probability. Thus, it follows that Π satisfies the following spectral approximation guarantee

$$\frac{\Phi^\top\Phi + \lambda I}{1+\epsilon} \preceq \Phi^\top\Pi^\top\Pi\Phi + \lambda I \preceq \frac{\Phi^\top\Phi + \lambda I}{1-\epsilon}.$$

Algorithm 1 invokes the RowSampler primitive $\log \frac{\|\Phi\|_F^2}{\epsilon\lambda} = O(\log n)$ times. Thus, by Lemma B.1, the runtime of finding Π is $O\left(\frac{q^7 \cdot s_\lambda^2 \cdot n}{\epsilon^4} \log^5 n \log q + \log^4 n \log q \cdot \sum_i \operatorname{nnz}\left(X^{(i)}\right)\right)$.

□

C. Proof of Lemma 4.1

First, by properties of tensor products and using the definitions of sketch matrices $S^{(c)} = \frac{1}{\sqrt{m}} \cdot P_c HD$, we obtain

$$\left(S^{(1)}X\right) \otimes \left(S^{(2)}X\right) \otimes \ldots \left(S^{(q)}X\right) = \frac{1}{m^{q/2}} \cdot (P_1 \times P_2 \times \ldots P_q) \cdot (HDX)^{\otimes q}, \tag{15}$$

where $P_1 \times P_2 \times \ldots P_q$ denotes the Kronecker product of the sampling matrices $P_1, P_2, \ldots P_q$ and is of size $m^q \times d^q$. Now let $x_1, x_2, \ldots x_n \in \mathbb{R}^d$ denote the columns of X. By Khintchine's inequality (Lemma A.1) along with a union bound over the d entries of the vector HDx_ℓ, the following holds with probability $1 - \frac{1}{\operatorname{poly}(n)}$, for every $\ell \in [n]$:

$$\|HD \cdot x_\ell\|_\infty^2 \leq O(\log n) \cdot \|x_\ell\|_2^2.$$

Therefore, using the definition of tensor product, the following holds with probability $1 - \frac{1}{\operatorname{poly}(n)}$, simultaneously for all $\ell \in [n]$ and all $r \in [q]$

$$\left\|(HD \cdot x_\ell)^{\otimes r}\right\|_\infty^2 \leq O(\log n)^r \cdot \left\|x_\ell^{\otimes r}\right\|_2^2. \tag{16}$$

From now on we condition on the above inequality holding for every $r \in [q]$ and every $\ell \in [n]$.

Now let us consider the matrix $(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top$. This matrix has d^q rows and n columns. If we let $\lambda_{\min}$ be the smallest eigenvalue of K, then using the properties of the tensor product of matrices, the Frobenius norm of this matrix satisfies the following inequality,

$$\begin{aligned}\left\|(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top\right\|_F^2 &= \left\|\left(\Sigma \otimes (HDX)^{\otimes q}\right) \cdot K^\top\right\|_F^2 \\ &\geq \lambda_{\min}^2 \cdot \left\|\Sigma \otimes (HDX)^{\otimes q}\right\|_F^2 = d^q \cdot \lambda_{\min}^2 \cdot \left\|\Sigma \otimes X^{\otimes q}\right\|_F^2 \end{aligned} \tag{17}$$

Furthermore, if we let $\lambda_{\max}$ be the largest eigenvalue of K, then for any row $\mathbf{j} \in [d]^q$ of the matrix $(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top$, the following upper bound holds,

$$\begin{aligned}\left\| \left[(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top \right]_{\mathbf{j},\star} \right\|_2^2 &= \left\| \Sigma \cdot \operatorname{diag}\left(\left[(HDX)^{\otimes q} \right]_{\mathbf{j},\star} \right) \cdot K^\top \right\|_F^2 \\ &\leq \lambda_{\max}^2 \cdot \left\| \Sigma \cdot \operatorname{diag}\left(\left[(HDX)^{\otimes q} \right]_{\mathbf{j},\star} \right) \right\|_F^2 \\ &= \lambda_{\max}^2 \cdot \sum_{\ell=1}^{n} \left| \left[(HD \cdot x_\ell)^{\otimes q} \right] (\mathbf{j}) \right|^2 \cdot \|\Sigma_{\star,\ell}\|_2^2\end{aligned}$$

By incorporating (16) into the above inequality for $r = q$, we find that for any $\mathbf{j} \in [d]^q$,

$$\left\| \left[(HDX)^{\otimes q} (\Sigma \otimes K)^\top \right]_{\mathbf{j},\star} \right\|_2^2 \leq O\left(\log n\right)^q \lambda_{\max}^2 \sum_{\ell=1}^{n} \left\| x_\ell^{\otimes q} \right\|_2^2 \|\Sigma_{\star,\ell}\|_2^2 = O\left(\log n\right)^q \cdot \lambda_{\max}^2 \left\| \Sigma \otimes X^{\otimes q} \right\|_F^2$$

In fact, we can prove a stronger version of the above inequality which will turn out to be very useful in our analysis. Let $\mathbf{j} \in [d]^q$ be some arbitrary index vector. Also, let $S \subseteq [q]$ be some arbitrary subset. Let us denote the subset of indices in $[d]^q$ that agree with $\mathbf{j}$ on S by $[d]^q_{\mathbf{j}_S}$ and formally define it as follows:

$$[d]^q_{\mathbf{j}_S} := \{\mathbf{i} \in [d]^q : \mathbf{i}_t = \mathbf{j}_t \text{ for all } t \in S\}.$$

Using this notation along with the properties of tensor products and (16) we have the following for every $\mathbf{j} \in [d]^q$ and $S \subseteq [q]$,

$$\begin{aligned}\sum_{\mathbf{i} \in [d]^q_{\mathbf{j}_S}} \left\| \left[(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top \right]_{\mathbf{i},\star} \right\|_2^2 &= \left\| (HDX)^{\otimes (q-|S|)} \cdot \prod_{t \in S} \operatorname{diag}\left([HDX]_{\mathbf{j}_t,\star} \right) \cdot (\Sigma \otimes K)^\top \right\|_F^2 \\ &\leq \lambda_{\max}^2 \cdot \left\| \left(\Sigma \otimes (HDX)^{\otimes (q-|S|)} \right) \prod_{t \in S} \operatorname{diag}\left([HDX]_{\mathbf{j}_t,\star} \right) \right\|_F^2 \\ &= \lambda_{\max}^2 \cdot \sum_{\ell \in [n]} \left\| \left(\Sigma_{\star,\ell} \otimes (HD \cdot x_\ell)^{\otimes (q-|S|)} \right) \cdot \prod_{t \in S} [HD \cdot x_\ell](\mathbf{j}_t) \right\|_2^2 \\ &= \lambda_{\max}^2 \cdot \sum_{\ell \in [n]} \left\| \Sigma_{\star,\ell} \otimes (HD \cdot x_\ell)^{\otimes (q-|S|)} \right\|_2^2 \cdot \prod_{t \in S} |[HD \cdot x_\ell](\mathbf{j}_t)|^2 \\ &\leq \lambda_{\max}^2 \cdot d^{q-|S|} \cdot \sum_{\ell \in [n]} \left\| \Sigma_{\star,\ell} \otimes x_\ell^{\otimes (q-|S|)} \right\|_2^2 \cdot \prod_{t \in S} O(\log n) \cdot \|x_\ell\|_2^2 \\ &= O(\log n)^{|S|} \cdot \lambda_{\max}^2 \cdot d^{q-|S|} \cdot \left\| \Sigma \otimes X^{\otimes q} \right\|_F^2,\end{aligned}$$

where the fifth line above follows from (16) for $r = 1$. Now by combining the above with (17) we find the following for every non-empty set $S \subseteq [q]$,

$$\max_{\mathbf{j} \in [d]^q} \left\{ \sum_{\mathbf{i} \in [d]^q_{\mathbf{j}_S}} \left\| \left[(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top \right]_{\mathbf{i},\star} \right\|_2^2 \right\} \leq O\left(\frac{\log n}{d} \right)^{|S|} \cdot \kappa^2 \cdot \left\| (HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top \right\|_F^2, \tag{18}$$

where $\kappa = \frac{\lambda_{\max}}{\lambda_{\min}}$ is the condition number of K. This inequality shows that the rows of $(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top$ are "flat" and the Frobenius norm of this matrix is spread-out evenly over the rows of this matrix. In addition to (18), we can prove a stronger inequality for the case of sets of cardinality one. Specifically, we prove a stronger version of (18) for any singleton set S, i.e., $|S| = 1$. We start by denoting the sole element of set S by $\tilde{s}$, i.e., $S = \{\tilde{s}\}$. So when $S = \{\tilde{s}\}$, using the definition

of $[d]^q_{\mathbf{j}_S}$ we have $[d]^q_{\mathbf{j}_S} = \{\mathbf{i} \in [d]^q : \mathbf{i}_{\tilde{s}} = \mathbf{j}_{\tilde{s}}\}$. Therefore, by properties of tensor products, we can write for any $\mathbf{j}_{\tilde{s}} \in [d]$:

$$\begin{aligned}\sum_{\mathbf{i}\in[d]^q_{\mathbf{j}_S}} \left\| \left[(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top\right]_{\mathbf{i},\star} \right\|_2^2 &= \left\| (HDX)^{\otimes(q-1)} \cdot \operatorname{diag}\left([HDX]_{\mathbf{j}_{\tilde{s}},\star}\right) (\Sigma \otimes K)^\top \right\|_F^2 \\ &= d^{q-1} \cdot \left\| X^{\otimes(q-1)} \cdot \operatorname{diag}\left([HDX]_{\mathbf{j}_{\tilde{s}},\star}\right) (\Sigma \otimes K)^\top \right\|_F^2 \\ &= d^{q-1} \cdot \left\| \left[HDX \cdot \left(\Sigma \otimes K \otimes X^{\otimes(q-1)}\right)^\top\right]_{\mathbf{j}_{\tilde{s}},\star} \right\|_2^2.\end{aligned}$$

Using the above inequality along with Khintchine's inequality from Lemma A.1, we find that the following holds for any $S = \{\tilde{s}\}$, with probability at least $1 - \frac{1}{\operatorname{poly}(n)}$,

$$\begin{aligned}\sum_{\mathbf{i}\in[d]^q_{\mathbf{j}_S}} \left\| \left[(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top\right]_{\mathbf{i},\star} \right\|_2^2 &\le O(\log n) \cdot d^{q-1} \cdot \left\| X \cdot \left(\Sigma \otimes K \otimes X^{\otimes(q-1)}\right)^\top \right\|_F^2 \\ &= O(\log n) \cdot d^{q-1} \cdot \left\| \left(X \otimes X^{\otimes(q-1)}\right) (\Sigma \otimes K)^\top \right\|_F^2 \\ &= O\left(\frac{\log n}{d}\right) \cdot \left\| (HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top \right\|_F^2\end{aligned}$$

Now using the above inequality and union bounding over all $\tilde{s} \in [d]$ and $\mathbf{j}_{\tilde{s}} \in [d]$, we can conclude that with probability at least $1 - \frac{1}{\operatorname{poly}(n)}$, the following holds simultaneously for all singleton sets $S = \{\tilde{s}\} \subseteq [q]$ and all $\mathbf{j} \in [d]^q$,

$$\max_{\mathbf{j}\in[d]^q} \left\{ \sum_{\mathbf{i}\in[d]^q_{\mathbf{j}_S}} \left\| \left[(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top\right]_{\mathbf{i},\star} \right\|_2^2 \right\} \le O\left(\frac{\log n}{d}\right) \cdot \left\| (HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top \right\|_F^2, \tag{19}$$

which is a stronger upper bound than (18) by a factor of κ^2.

Now recall that, by (15), we have the following,

$$\left\| \left(S^{(1)}X\right) \otimes \ldots \left(S^{(q)}X\right) \cdot (\Sigma \otimes K)^\top \right\|_F^2 = \frac{1}{m^q} \cdot \left\| (P_1 \times P_2 \times \ldots P_q) \cdot (HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top \right\|_F^2$$

Therefore, to simplify the notation, if we denote the vector corresponding to row norms of $(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top$ by $y \in \mathbb{R}^{d^q}$,

$$y_{\mathbf{j}} := \left\| \left[(HDX)^{\otimes q} \cdot (\Sigma \otimes K)^\top\right]_{\mathbf{j},\star} \right\|_2 \qquad \text{for every } \mathbf{j} \in [d]^q,$$

then it suffices to prove that

$$\Pr_{P_1,\ldots P_q}\left[\frac{1}{m^q} \cdot \|(P_1 \times P_2 \times \ldots P_q) \cdot y\|_2^2 \in (1 \pm \epsilon)\frac{\|y\|_2^2}{d^q}\right] \ge 1 - \delta \tag{20}$$

given the fact that the P_i are independent random sampling matrices and conditioned on y satisfying the following flatness property for any non-empty set $S \subseteq [q]$ (by combining (18) and (19)):

$$\max_{\mathbf{j}\in[d]^q} \sum_{\mathbf{i}\in[d]^q_{\mathbf{j}_S}} |y_{\mathbf{i}}|^2 \le O\left(\frac{\log n}{d}\right)^{|S|} \cdot \left(\kappa^2 \cdot \mathbb{1}_{\{|S|>1\}} + \mathbb{1}_{\{|S|=1\}}\right) \cdot \|y\|_2^2. \tag{21}$$

In order to prove (20), first note that $\frac{1}{m^q} \cdot \|(P_1 \times P_2 \times \dots P_q) \cdot y\|_2^2$ is an unbiased estimator, i.e.,

$$\begin{aligned}
\mathbb{E}_{P_1,\dots P_q}\left[\frac{1}{m^q} \cdot \|(P_1 \times \dots P_q) \cdot y\|_2^2\right] &= \frac{1}{m^q} \cdot \sum_{i_1,i_2,\dots i_q \in [d]} \Pr[i_1 \in P_1] \cdot \dots \Pr[i_q \in P_q] \cdot |y_{(i_1,\dots i_q)}|^2 \\
&= \frac{1}{d^q} \cdot \sum_{i_1,i_2,\dots i_q \in [d]} |y_{(i_1,\dots i_q)}|^2 \\
&= \frac{\|y\|_2^2}{d^q},
\end{aligned}$$

where by $\Pr[i_c \in P_c]$ we mean the probability that i_c is sampled by matrix P_c, and this quantity is equal to $\Pr[i_c \in P_c] \equiv \frac{m}{d}$. Next we bound the variance of this estimator and then finish the proof by Chebyshev's inequality.

$$\begin{aligned}
&\mathbb{E}_{P_1,\dots P_q}\left[\left(\frac{1}{m^q} \cdot \|(P_1 \times P_2 \times \dots P_q) \cdot y\|_2^2\right)^2\right] \\
&= \frac{1}{m^{2q}} \sum_{\mathbf{j},\mathbf{i} \in [d]^q} \Pr[\mathbf{i}_1, \mathbf{j}_1 \in P_1] \cdot \dots \Pr[\mathbf{i}_q, \mathbf{j}_q \in P_q] \cdot |y_\mathbf{i}|^2 \cdot |y_\mathbf{j}|^2 \\
&= \frac{1}{m^{2q}} \sum_{S \subseteq [q]} \sum_{\mathbf{j} \in [d]^q} \sum_{\mathbf{i} \in [d]^q_{\mathbf{j}_S}} \left(\prod_{t \in [q] \setminus S} \mathbb{1}_{\{\mathbf{j}_t \neq \mathbf{i}_t\}}\right) \cdot \Pr[\mathbf{i}_1, \mathbf{j}_1 \in P_1] \cdot \dots \Pr[\mathbf{i}_q, \mathbf{j}_q \in P_q] \cdot |y_\mathbf{i}|^2 \cdot |y_\mathbf{j}|^2 \\
&= \frac{1}{m^{2q}} \sum_{S \subseteq [q]} \sum_{\mathbf{j} \in [d]^q} \sum_{\mathbf{i} \in [d]^q_{\mathbf{j}_S}} \left(\prod_{t \in [q] \setminus S} \mathbb{1}_{\{\mathbf{j}_t \neq \mathbf{i}_t\}} \cdot \Pr[\mathbf{i}_t, \mathbf{j}_t \in P_t]\right) \cdot \left(\prod_{t' \in S} \Pr[\mathbf{j}_{t'} \in P_{t'}]\right) \cdot |y_\mathbf{i}|^2 \cdot |y_\mathbf{j}|^2 \qquad (22)
\end{aligned}$$

Where the second line follows because $P_1, \dots P_q$ are independent and the third line follows from the definition of the set $[d]^q_{\mathbf{j}_S}$. Now we can bound (22) by noting that for any $i \neq j$, the collision probability $\Pr[i, j \in P_t] = \frac{m(m-1)}{d(d-1)} \leq \left(\frac{m}{d}\right)^2$ and $\Pr[j \in P_t] = \frac{m}{d}$. We can write,

$$\begin{aligned}
\mathbb{E}_{P_1,\dots P_q}\left[\left(\frac{1}{m^q} \cdot \|(P_1 \times P_2 \times \dots P_q) \cdot y\|_2^2\right)^2\right] &\leq \frac{1}{m^{2q}} \sum_{\mathbf{j} \in [d]^q} \sum_{\mathbf{i} \in [d]^q} \left(\frac{m}{d}\right)^{2q} \cdot |y_\mathbf{i}|^2 \cdot |y_\mathbf{j}|^2 \\
&\quad + \frac{1}{m^{2q}} \sum_{\emptyset \neq S \subseteq [q]} \sum_{\mathbf{j} \in [d]^q} \sum_{\mathbf{i} \in [d]^q_{\mathbf{j}_S}} \left(\frac{m}{d}\right)^{2q-|S|} \cdot |y_\mathbf{i}|^2 \cdot |y_\mathbf{j}|^2 \\
&= \frac{\|y\|_2^4}{d^{2q}} + \sum_{\emptyset \neq S \subseteq [q]} \frac{1}{m^{|S|} \cdot d^{2q-|S|}} \sum_{\mathbf{j} \in [d]^q} |y_\mathbf{j}|^2 \sum_{\mathbf{i} \in [d]^q_{\mathbf{j}_S}} |y_\mathbf{i}|^2 \\
&\leq \frac{\|y\|_2^4}{d^{2q}} + \sum_{\substack{S \subseteq [q] \\ |S|=1}} \frac{O(\log n) \|y\|_2^2}{m \cdot d^{2q}} \sum_{\mathbf{j} \in [d]^q} |y_\mathbf{j}|^2 \\
&\quad + \sum_{\substack{S \subseteq [q] \\ |S|>1}} \frac{O(\log n)^{|S|} \cdot \kappa^2 \|y\|_2^2}{m^{|S|} \cdot d^{2q}} \sum_{\mathbf{j} \in [d]^q} |y_\mathbf{j}|^2 \\
&\leq \frac{\|y\|_2^4}{d^{2q}} + O\left(\frac{q \log n}{m} + \frac{q^2 \kappa^2 \log^2 n}{m^2}\right) \cdot \frac{\|y\|_2^4}{d^{2q}},
\end{aligned}$$

where the fourth and fifth lines above follow from the fact that y satisfies the condition in (21). Therefore, the above inequality along with the fact that $\frac{1}{m^q} \cdot \|(P_1 \times P_2 \times \dots P_q) \cdot y\|_2^2$ is an unbiased estimator implies that,

$$\mathrm{Var}_{P_1,\dots P_q}\left[\frac{1}{m^q} \cdot \|(P_1 \times P_2 \times \dots P_q) \cdot y\|_2^2\right] = O\left(\frac{q \log n}{m} + \frac{q^2 \kappa^2 \log^2 n}{m^2}\right) \cdot \frac{\|y\|_2^4}{d^{2q}}$$

Thus if $m = C\left(\frac{1}{\epsilon^2} + \frac{\kappa}{\epsilon}\right) \cdot \frac{q}{\delta} \log n$ for a large enough constant C, by using the definition of vector y together with Chebyshev's inequality and a union bound, we have the following,

$$\Pr\left[\left\|\left(\left(S^{(1)}X\right) \otimes \ldots \left(S^{(q)}X\right)\right) \cdot (\Sigma \otimes K)^\top\right\|_F^2 \in (1 \pm \epsilon)\left\|X^{\otimes q} \cdot (\Sigma \otimes K)^\top\right\|_F^2\right] \geq 1 - \delta,$$

so the lemma statement follows.

The runtime of applying all sketches to X consists of the time to compute $Y = HDX$ and the time to compute $P_r Y$ for every $r \in [q]$. The time to compute Y is $O(nd \log d)$ by using the FFT algorithm and the time to compute all $P_r Y$ matrices is $O(qmn)$.

D. Leverage Score Sampler for Polynomial Kernel

D.1. Proof of Lemma 4.2

All rows of the sampling matrix $S \in \mathbb{R}^{s \times d^q}$ (the output of Algorithm 3) have independent and identical distributions because for each $\ell \in [s]$, the ℓ^{th} row of S is constructed by sampling indices $i_1, i_2, \cdots i_q$ in line 19 completely independent of the sampled values for other rows $\ell' \neq \ell$. Thus, it is enough to consider the distribution of the ℓ^{th} row of S for some arbitrary $\ell \in [s]$. Let $I := (I_1, \ldots I_q)$ be a vector-valued random variable that takes values in $[d]^q$ with the following conditional probability distribution for every $a = 1, 2, \cdots q$ and every $i \in [d]$,

$$\Pr\left[I_a = i | I_1 = i_1, \cdots I_{a-1} = i_{a-1}\right] := p^a_{h(i)} \cdot q^a_i, \tag{23}$$

where distributions $\{p^a_r\}_{r \in [s']}$ and $\{q^a_i\}_{i \in h^{-1}(t)}$ for every $t \in [s']$ are defined as per lines 15 and 18 of the algorithm. One can verify that the random vector $(i_1, i_2, \cdots i_q)$ obtained by stitching together the random indices generated in line 19 of the algorithm, is in fact a copy of I defined above.

Let β_ℓ be the quantity computed in line 21 of the algorithm. If $i_1, i_2, \cdots i_q \in [d]$ are the indices sampled in line 19 of the algorithm, then using the conditional distribution of I in (23), we find that the value of β_ℓ is equal to the following,

$$\begin{aligned}
\beta_\ell &= s \cdot \prod_{a=1}^{q} p^a_{h(i_a)} q^a_{i_a} \\
&= s \cdot \prod_{a=1}^{q} \Pr\left[I_a = i_a | I_1 = i_1, \cdots I_{a-1} = i_{a-1}\right] \\
&= s \cdot \Pr\left[I = (i_1, i_2, \ldots i_q)\right],
\end{aligned}$$

where p^a and q^a are the distributions computed in lines 15 and 18 of the algorithm. Hence, for any $i_1, i_2, \cdots i_q \in [d]$, the distribution of $S_{\ell,\star}$ is,

$$\Pr\left[S_{\ell,\star} = \beta_\ell^{-1/2}(e_{i_1} \otimes e_{i_2} \otimes \cdots e_{i_q})^\top\right] = \Pr\left[I = (i_1, i_2, \ldots i_q)\right] = \frac{\beta_\ell}{s}. \tag{24}$$

Now to ease the notation we define $Y^{(c)}_k := \bigotimes_{j=c}^{q} S^{(j)}_k X$ for every $k \in [m']$ and $c \in [q]$, where $S^{(c)}_k$ are the SRHT sketches with shared signs drawn in line 4 of the algorithm. From the definition of $\text{TN}^{(k)}$ in line 5 and by invoking Lemma 3.1 we have the following inequalities for any $r \in [s']$, $k \in [m']$, $i \in [d]$, and any $a = 1, 2, \ldots q$, with probability at least $1 - \frac{1}{\text{poly}(n)}$,

$$\text{TN}^{(k)}.\textsc{Query}\left(L^a_{r,k}, a\right) \in \left(1 \pm \frac{1}{40q}\right)\left\|\left(Y^{(a+1)}_k \otimes M\right) D^a W^\top_{r,k}\right\|_F^2, \tag{25}$$

$$\text{TN}^{(k)}.\textsc{Query}\left(D^a X^\top_{i,\star}, a\right) \in \left(1 \pm \frac{1}{40q}\right)\left\|\left(Y^{(a+1)}_k \otimes M\right) D^a X^\top_{i,\star}\right\|_2^2 \tag{26}$$

By union bounding over $qds'm'$ events, (25) and (26) hold simultaneously for all $a \in [q]$, $k \in [m']$, $i \in [d]$, and all $r \in [s']$ with high probability. From now on we condition on (25) and (26).

Furthermore, note that $W_{r,k}$ is defined in line 9 as $W_{r,k} = G_r^k \cdot X_{h^{-1}(r),\star}$, where G_r^k is a degree-1 PolySketch with target dimension $n' = C_3 q^2$. By Lemma 2.3, G_r^k approximately preserves Frobenius norm of any fixed matrix with constant probability. In particular, for every $a \in [q], r \in [s'], k \in [m']$, with probability at least $19/20$:

$$\left\| \left(Y_k^{(a+1)} \otimes M \right) D^a W_{r,k}^\top \right\|_F^2 \in \left(1 \pm \frac{1}{80q} \right) \left\| Y_k^{(a+1)} D^a \left(X_{h^{-1}(r),\star} \otimes M \right)^\top \right\|_F^2. \tag{27}$$

To obtain the above inequality we used the fact that there is a bijective correspondence between entries of $\left(Y_k^{(a+1)} \otimes M \right) D^a X_{h^{-1}(r),\star}^\top$ and $Y_k^{(a+1)} D^a \left(X_{h^{-1}(r),\star} \otimes M \right)^\top$.

Additionally, note that $M = H \cdot (B^\top B + \lambda I)^{-1/2}$ for a random Gaussian matrix H with $d' = \Omega(q^2 \log n)$ rows. Therefore, H is a JL-transform. So if we define $A := (B^\top B + \lambda I)^{-1/2}$ for ease of notation, then with probability $1 - \frac{1}{\mathrm{poly}(n)}$, the following holds for any $a \in [q], r \in [s']$:

$$\left\| Y_k^{(a+1)} D^a \left(X_{h^{-1}(r),\star} \otimes M \right)^\top \right\|_F^2 \in \left(1 \pm \frac{1}{80q} \right) \left\| Y_k^{(a+1)} D^a \left(X_{h^{-1}(r),\star} \otimes A \right)^\top \right\|_F^2.$$

By union bounding over qs' events we can conclude that the above inequality holds simultaneously for all $a \in [q]$ and $r \in [s']$. From now on we condition on the above inequality holding. By combining this condition with (27) we find that with probability at least $19/20$ the following holds:

$$\left\| \left(Y_k^{(a+1)} \otimes M \right) D^a W_{r,k}^\top \right\|_F^2 \in \left(1 \pm \frac{1}{39q} \right) \left\| Y_k^{(a+1)} D^a \left(X_{h^{-1}(r),\star} \otimes A \right)^\top \right\|_F^2. \tag{28}$$

Using the definition of matrices $Y_k^{(c)} := \bigotimes_{j=c}^q S_k^{(j)} X$ and by Lemma 4.1, because the number of rows of $S_k^{(c)}$ is $m'' = \Omega(q^3 + q^2 \kappa \log n)$, the following holds with probability at least $19/20$ for any $a \in [q], r \in [s'], k \in [m']$,

$$\left\| Y_k^{(a+1)} D^a \left(X_{h^{-1}(r),\star} \otimes A \right)^\top \right\|_F^2 \in \left(1 \pm \frac{1}{80q} \right) \left\| X^{\otimes(q-a)} D^a \left(X_{h^{-1}(r),\star} \otimes A \right)^\top \right\|_F^2.$$

By combining the above with (28) using a union bound, and plugging the result into (25) we find that with probability at least $9/10$ the following holds,

$$\mathrm{TN}^{(k)}.\textsc{Query}\left(L_{r,k}^a, a \right) \in \left(1 \pm \frac{1}{10q} \right) \left\| X^{\otimes(q-a)} \cdot D^a \left(X_{h^{-1}(r),\star} \otimes A \right)^\top \right\|_F^2.$$

By taking the median of $m' = \Omega(\log n)$ independent instances of $\mathrm{TN}^{(k)}.\textsc{Query}\left(L_{r,k}^a, a \right)$, the success probability of the above gets boosted. Thus, by a union bound, with probability at least $1 - \frac{1}{\mathrm{poly}(n)}$ the following holds simultaneously for all $a \in [q]$ and $r \in [s']$,

$$\textsc{Median}_{k \in [m']} \left\{ \mathrm{TN}^{(k)}.\textsc{Query}\left(L_{r,k}^a, a \right) \right\} \in \left(1 \pm \frac{1}{10q} \right) \left\| X^{\otimes(q-a)} D^a \left(X_{h^{-1}(r),\star} \otimes A \right)^\top \right\|_F^2 \tag{29}$$

Similarly, we can use the fact that there is a bijective correspondence between the entries of $\left(Y_k^{(a+1)} \otimes M \right) D^a X_{i,\star}^\top$ and $Y_k^{(a+1)} D^a \mathrm{diag}\left(X_{i,\star} \right) M^\top$ along with $M = H \cdot A$ to conclude that with probability $1 - \frac{1}{\mathrm{poly}(n)}$, the following holds for any $a \in [q], r \in [s'], k \in [m'], i \in [d]$:

$$\left\| \left(Y_k^{(a+1)} \otimes M \right) D^a X_{i,\star}^\top \right\|_2^2 \in \left(1 \pm \frac{1}{80q} \right) \left\| Y_k^{(a+1)} D^a \cdot \mathrm{diag}\left(X_{i,\star} \right) A \right\|_F^2 \tag{30}$$

By a union bound over $qs'm'd$ events we can conclude that the above inequality holds simultaneously for all $a \in [q], r \in [s'], k \in [m'], i \in [d]$. From now on we condition on the above inequality holding. Then, by using the definition of matrices $Y_k^{(c)} := \bigotimes_{j=c}^q S_k^{(j)} X$ and invoking Lemma 4.1, the following holds with probability at least $19/20$ for any $a \in [q], r \in [s'], k \in [m'], i \in [d]$,

$$\left\| Y_k^{(a+1)} D^a \mathrm{diag}\left(X_{i,\star} \right) A \right\|_F^2 \in \left(1 \pm \frac{1}{80q} \right) \left\| X^{\otimes(q-a)} \cdot D^a \mathrm{diag}\left(X_{i,\star} \right) A \right\|_F^2$$

By combining this with the condition in (30) and (26) we find that with probability at least $19/20$:

$$\text{TN}^{(k)}.\textsc{Query}\left(D^a X_{i,\star}^\top, a\right) \in \left(1 \pm \frac{1}{15q}\right) \left\|X^{\otimes(q-a)} D^a \text{diag}\left(X_{i,\star}\right) A\right\|_F^2$$

By taking the median of $m' = \Omega(\log n)$ independent instances of $\text{TN}^{(k)}.\textsc{Query}\left(D^a X_{i,\star}^\top, a\right)$, the success probability of the above gets boosted. Thus, by applying the median trick and then using a union bound, with probability at least $1 - \frac{1}{\text{poly}(n)}$ the following holds simultaneously for all $a \in [q], i \in [d]$ and $r \in [s']$,

$$\textsc{Median}_{k\in[m']}\left\{\text{TN}^{(k)}.\textsc{Query}\left(D^a X_{i,\star}^\top, a\right)\right\} \in \left(1 \pm \frac{1}{15q}\right) \left\|X^{\otimes(q-a)} D^a \text{diag}\left(X_{i,\star}\right) A\right\|_F^2$$

Plugging the above inequality along with (29) into (23), we conclude that with high probability the following bound holds simultaneously for all $a \in [q]$ and all $i \in [d]$,

$$\Pr[I_a = i | I_1 = i_1, I_2 = i_2, \cdots I_{a-1} = i_{a-1}] \geq \left(1 - \frac{1}{3q}\right) \cdot \frac{\left\|X^{\otimes(q-a)} D^a \cdot \text{diag}\left(X_{i,\star}\right) A\right\|_F^2}{\left\|X^{\otimes(q-a+1)} D^a A\right\|_F^2}. \tag{31}$$

Thus, using the definition of D^a and $A = (B^\top B + \lambda I)^{-1/2}$, we have

$$\begin{aligned}
\Pr\left[I = (i_1, i_2, \cdots i_q)\right] &= \prod_{a=1}^{q} \Pr\left[I_a = i_a | I_1 = i_1, \cdots I_{a-1} = i_{a-1}\right] \\
&\geq \prod_{a=1}^{q} \left(1 - \frac{1}{3q}\right) \frac{\left\|X^{\otimes(q-a)} D^a \cdot \text{diag}\left(X_{i_a,\star}\right) A\right\|_F^2}{\left\|X^{\otimes(q-a+1)} D^a A\right\|_F^2} \\
&\geq \frac{1}{2} \cdot \frac{\left\|\mathbf{1}_n^\top \cdot D^q \cdot \text{diag}\left(X_{i_q,\star}\right) A\right\|_F^2}{\|X^{\otimes q} D^1 A\|_F^2} \\
&= \frac{1}{2} \cdot \frac{\left\|\left[X^{\otimes q} \cdot (B^\top B + \lambda I)^{-1/2}\right]_{(i_1,i_2,\cdots i_q),\star}\right\|_2^2}{\left\|X^{\otimes q} \cdot (B^\top B + \lambda I)^{-1/2}\right\|_F^2}
\end{aligned}$$

This shows that, with high probability in n,

$$\Pr\left[S_{\ell,\star} = \beta_\ell^{-1/2} (e_{i_1} \otimes e_{i_2} \otimes \cdots e_{i_q})^\top\right] \geq \frac{1}{2} \cdot \frac{\left\|\left[X^{\otimes q} \cdot (B^\top B + \lambda I)^{-1/2}\right]_{(i_1,i_2,\cdots i_q),\star}\right\|_2^2}{\left\|X^{\otimes q} \cdot (B^\top B + \lambda I)^{-1/2}\right\|_F^2}$$

Because $\frac{\beta_\ell}{s}$ is the probability of sampling row $(i_1, i_2, \cdots i_q)$ of $X^{\otimes q}(B^\top B + \lambda I)^{-1/2}$, the above inequality proves that with high probability, matrix S is a rank-s row norm sampler for $X^{\otimes q}(B^\top B + \lambda I)^{-1/2}$ as in Definition 2.5.

Runtime: The first expensive step of this algorithm is the computation of M in line 3 which takes $O(m^2 n + q^2 mn \log n)$ operations since B has rank at most m. The next expensive computation is the computation of $S_k^{(c)} X$ for $c \in [q]$ and $k \in [m']$ in line 5 of the algorithm. By Lemma 4.1, the total time to compute these sketched matrices is $O\left((q^4 + q^3\kappa) n \log^2 n + nd \log^2 n\right)$. Another expensive step is the construction of the $\text{TN}^{(k)}$ data-structure in line 5 for $k \in [m']$. By Lemma 3.1, these DS's for $\epsilon = \frac{1}{40q}$ and all $k \in [m']$ can be formed in total time $O\left(q^4 \log^2 q \cdot n \log^2 n + (q^4 + q^3\kappa) n \log^3 n\right)$.

By Lemma 2.3, matrices $W_{r,k}$ for all $r \in [s']$ and $k \in [m']$ in line 9 of the algorithm can be computed in total time $O\left(q^2 s' n \log^2 n + \log n \cdot \text{nnz}\left(X\right)\right)$.

The matrix $W_{r,k}$ for every $k \in [m']$, and $r \in [s']$, has size $O(q^2) \times n$. Thus, by Lemma 3.1, computing the distribution $\{p_r^a\}_{r=1}^{s'}$ in line 15 takes time $O\left(q^4 s' \cdot n \log^2 n \log q\right)$ for a fixed $a \in [q]$ and a fixed $\ell \in [s]$. Therefore, the total time to compute this distribution for all a and ℓ is $O\left(q^8 s^2 \cdot n \log^2 n \log q\right)$.

The runtime of computing the distribution $\{q_i^a\}_{i\in h^{-1}(t)}$ in line 18 depends on the sparsity of $X_{h^{-1}(t),\star}$, i.e., nnz $\left(X_{h^{-1}(t),\star}\right)$. To bound the sparsity of $X_{h^{-1}(t),\star}$, note that, the hash function h is fully independent. Thus, by invoking Bernstein's inequality, we find that, $\Pr\left[\text{nnz}\left(X_{h^{-1}(t),\star}\right) = O\left(\log n \cdot \left(\sqrt{\frac{n}{s'}\text{nnz}\left(X\right)} + n\right)\right)\right] \geq 1 - \frac{1}{\text{poly}(n)}$. Hence, by union bounding over qs' events, with high probability in n, nnz $\left(X_{h^{-1}(t),\star}\right) = O\left(\left(\text{nnz}\left(X\right)/s' + n\right)\log n\right)$, simultaneously for all $t \in [s']$ and $a \in [q]$.

Therefore, by Lemma 3.1, the distribution $\{q_i^a\}_{i\in h^{-1}(t)}$ in line 18 of the algorithm can be computed in total time $O\left(q^3 sn\log^4 n\log q + \log^4 n\log q\cdot\text{nnz}\left(X\right)\right)$ for all $a\in[q]$ and all $\ell\in[s]$.

The total runtime of Algorithm 3 is thus $O(m^2n + q^8s^2n\log^2 n\log q + q^3\kappa n\log^3 n + dn\log^4 n)$.

D.2. Proof of Theorem 4.3

The theorem follows by invoking Lemmas 2.6 and 4.2. To find the sampling matrix Π, run Algorithm 1 on Φ with $\mu = s_\lambda$ and for the RowSampler primitive, invoke Algorithm 3. By Lemma 4.2, Algorithm 3 outputs a row norm sampler as per Definition 2.5, with probability $1 - \frac{1}{\text{poly}(n)}$. Therefore, since the total number of times Algorithm 3 is invoked by Algorithm 1 is $\log\frac{\|\Phi\|_F^2}{\epsilon\lambda} = O(\log n)$, by a union bound, the preconditions of Lemma 2.6 are satisfied with high probability. Thus, it follows that Π satisfies the following spectral approximation guarantee

$$\frac{\Phi^\top\Phi + \lambda I}{1+\epsilon} \preceq \Phi^\top\Pi^\top\Pi\Phi + \lambda I \preceq \frac{\Phi^\top\Phi+\lambda I}{1-\epsilon}.$$

The only thing that remains is bounding the runtime. In the proof of Lemma 2.6 in (Woodruff & Zandieh, 2020), it is shown that with high probability at any iteration $t\in[T]$ of Algorithm 1, the following holds,

$$\frac{\Phi^\top\Phi + \lambda_t I}{1+\epsilon} \preceq \Phi^\top\Pi_t^\top\Pi_t\Phi + \lambda_t I \preceq \frac{\Phi^\top\Phi+\lambda_t I}{1-\epsilon}.$$

Therefore, $\left\|\Phi^\top\Pi_t^\top\Pi_t\Phi\right\| = O(\|\Phi^\top\Phi\|)$. Now note that Algorithm 1 invokes the RowSampler primitive $T = \log\frac{\|\Phi\|_F^2}{\epsilon\lambda} = O(\log n)$ times. Thus, by Lemma 4.2, the runtime of finding Π is the sum of $O\left(\frac{q^8s_\lambda^2 n\log^4 n}{\epsilon^4} + \sqrt{\frac{\left\|\Phi^\top\Pi_t^\top\Pi_t\Phi\right\|}{\lambda_t}}q^3 n\log^3 n + nd\log^4 n\right)$ for all $t\in[T]$. Since $\lambda_t = 2^{T-t}\lambda$ has a geometric decay, the total time complexity is $O\left(\frac{q^8s_\lambda^2 n\log^5 n}{\epsilon^4} + \sqrt{\frac{\|\Phi^\top\Phi\|}{\lambda}}q^3 n\log^3 n + nd\log^5 n\right)$.

E. Spectral Approximation to Generalized Polynomial Kernels

In this section we design an algorithm that can produce a spectral approximation to the GPK defined in Definition 5.1. Our approach is to perform leverage score sampling on the GPK feature matrix Φ defined in (4). We do this by invoking our recursive sampling method given in Algorithm 1 on Φ. Our central contribution is the design of RowSampler algorithm for the GPK feature matrix Φ that runs in input sparsity time. This procedure can perform *row norm sampling* as per Definition 2.5 on $\Phi(B^\top B+\lambda I)^{-1/2}$ for $\Phi = \bigoplus_{j=0}^q \alpha_j X^{\otimes j}\text{diag}(v)$ using $\widetilde{O}\left(\text{nnz}(X)\right)$ runtime. Our primitive is an adaptation and generalization of Algorithm 3.

The formal guarantee on Algorithm 5 is given in the following lemma.

Lemma E.1. *For any matrix $X\in\mathbb{R}^{d\times n}$, any vector $v\in\mathbb{R}^n$, any positive integers q, s, and any $\alpha\in\mathbb{R}^{q+1}$, let Φ be the GPK feature matrix defined in (4). For any matrix $B\in\mathbb{R}^{m\times n}$ and any $\lambda > 0$, with probability at least $1-\frac{1}{\text{poly}(n)}$, Algorithm 5 outputs a rank-s row norm sampler for $\Phi(B^\top B+\lambda I)^{-1/2}$ as per Definition 2.5, in time $O\left(m^2n + q^8s^2n\log^3 n + q^3\kappa n\log^3 n + nd\log^4 n\right)$, where $\kappa = \sqrt{\|B^\top B\|/\lambda + 1}$.*

Proof. All rows of the sampling matrix $S\in\mathbb{R}^{s\times d^q}$ (the output of Algorithm 3) have independent and identical distributions because for each $\ell\in[s]$, the ℓ^{th} row of the matrix S is constructed by sampling the degree b and indices $i_1, i_2, \cdots i_q$ in lines13 and 22, respectively, completely independent of the sampled values for other rows $\ell'\neq\ell$. Thus, it is enough to consider the distribution of the ℓ^{th} row of S for some arbitrary $\ell\in[s]$.

Algorithm 5 ROWSAMPLER for GPK features $\Phi = \bigoplus_{j=0}^{q} \alpha_j X^{\otimes j} \mathrm{diag}(v)$

input: $q, s \in \mathbb{Z}_+$, $X \in \mathbb{R}^{d \times n}$, $v \in \mathbb{R}^n$, $\alpha \in \mathbb{R}^{q+1}$, $B \in \mathbb{R}^{m \times n}$, $\lambda > 0$
output: Sampling matrix $S \in \mathbb{R}^{s \times d^q}$

1: $\kappa \leftarrow \sqrt{\frac{\|B^\top B\|}{\lambda} + 1}$
2: Generate $H \in \mathbb{R}^{d' \times n}$ with i.i.d. normal entries with $d' = C_0 q^2 \log n$ rows
3: $M \leftarrow H \cdot (B^\top B + \lambda I)^{-1/2}$
4: For every $k \in [m']$, let $S_k^{(1)}, S_k^{(2)}, \dots S_k^{(q)} \in \mathbb{R}^{m'' \times d}$ be independent copies of SRHT sketches with shared signs as per Lemma 4.1, where $m' = C_1 \log n$ and $m'' = C_2(q^3 + q^2\kappa)\log n$
5: For every $k \in [m']$, let $\mathrm{TNORM}^{(k)}$ be the DS in Algorithm 2 for inputs $\left(S_k^{(1)} X, \dots S_k^{(q)} X, M\right)$ and $\epsilon = \frac{1}{40q}$
6: Let $h : [d] \to [s']$ be a fully independent and uniform hash function with $s' = \lceil q^3 s \rceil$ buckets
7: Let $h^{-1}(r) = \{j \in [d] : h(j) = r\}$ for every $r \in [s']$
8: For every $r \in [s']$ and $k \in [m']$, let $G_r^k \in \mathbb{R}^{n' \times d_r}$ be independent instances of degree-1 POLYSKETCH as per Lemma 2.3, where $d_r = |h^{-1}(r)|$, $n' = C_3 q^2$
9: $W_{r,k} \leftarrow G_r^k \cdot X_{h^{-1}(r),\star}$ for every $k \in [m']$ and $r \in [s']$
10: $f_j \leftarrow \alpha_j^2 \cdot \mathrm{MEDIAN}_{k \in [m']} \mathrm{TNORM}^{(k)}.\mathrm{QUERY}(v, q-j)$ for every $j = 0, 1, \dots q$
11: $f_j \leftarrow f_j / \sum_{i=0}^{q} f_i$ for every $j = 0, 1, \dots q$
12: **for** $\ell = 1$ to s **do**
13: Sample $b \in \{0, 1, \dots q\}$ from distribution $\{f_j\}_{j=0}^{q}$
14: $D^1 \leftarrow \mathrm{diag}(v)$ and $\beta_\ell \leftarrow s \cdot f_b$
15: **for** $a = 1$ to b **do**
16: $L_{r,k}^a \leftarrow D^a \cdot W_{r,k}^\top$ for every $k \in [m']$, and $r \in [s']$
17: $p_r^a \leftarrow \mathrm{MEDIAN}_{k \in [m']} \mathrm{TNORM}^{(k)}.\mathrm{QUERY}(L_{r,k}^a, a + q - b)$ for every $r \in [s']$
18: $p_r^a \leftarrow p_r^a / \sum_{t=1}^{s'} p_t^a$ for every $r \in [s']$
19: Sample $t \in [s']$ from distribution $\{p_r^a\}_{r=1}^{s'}$
20: Let $q_i^a \leftarrow \mathrm{MEDIAN}_{k \in [m']} \mathrm{TNORM}^{(k)}.\mathrm{QUERY}(D^a X_{i,\star}^\top, a + q - b)$ for every $i \in h^{-1}(t)$
21: $q_i^a \leftarrow q_i^a / \sum_{j \in h^{-1}(t)} q_j^a$ for every $i \in h^{-1}(t)$
22: Sample $i_a \in [d]$ from distribution $\{q_i^a\}_{i \in h^{-1}(t)}$
23: $D^{a+1} \leftarrow D^a \cdot \mathrm{diag}\left(X_{i_a,\star}^{(a)}\right)$
24: $\beta_\ell \leftarrow \beta_\ell \cdot p_t^a q_{i_a}^a$
25: **if** $b > 0$ **then**
26: Let ℓ^{th} row of S be $\beta_\ell^{-1/2}\big(\underbrace{0, 0, \dots 0}_{\frac{d^b - 1}{d-1} \text{ zeros}}, e_{i_1} \otimes e_{i_2} \otimes \cdots e_{i_b}, \underbrace{0, 0, \dots 0}_{\frac{d^{q+1} - d^{b+1}}{d-1} \text{ zeros}}\big)$
27: **else**
28: Let ℓ^{th} row of S be $\beta_\ell^{-1/2}\big(1, \underbrace{0, 0, \dots 0}_{\frac{d^{q+1} - d}{d-1} \text{ zeros}}\big)$
29: **return** S

Let U be a random variable that takes values in $\{0, 1, \dots q\}$ with the following distribution for every $a = 0, 1, \dots q$

$$\Pr[U = a] = f_a \tag{32}$$

where $\{f_a\}_{a=0}^{q}$ is the distribution defined in line 11 of the algorithm. Additionally, for any $b \in \{1, \dots q\}$, let $I^b := (I_1, \dots I_b)$ be a vector-valued random variable that takes values in $[d]^b$ with the following conditional probability distribution for every $a = 1, 2, \cdots b$ and every $i \in [d]$,

$$\Pr\left[I_a = i | I_1 = i_1, \cdots I_{a-1} = i_{a-1}\right] := p_{h(i)}^a \cdot q_i^a, \tag{33}$$

where distributions $\{p_r^a\}_{r \in [s']}$ and $\{q_i^a\}_{i \in h^{-1}(t)}$ for every $t \in [s']$ are defined as per lines 18 and 21 of the algorithm. One can verify that conditioned on Algorithm 5 sampling some $b = 1, \dots q$ in line 13, the random vector $(i_1, i_2, \cdots i_b)$ obtained

by stitching together the random indices generated in line 22 of the algorithm, is in fact a copy of I^b defined above. Note that if the algorithm samples degree $b = 0$ in line 13 then the algorithm does not sample any indices in line 22.

Let β_ℓ be the quantity computed in line 24 of the algorithm. If $b \in \{1, \dots q\}$ is the degree sampled in line 13 and $i_1, i_2, \cdots i_b \in [d]$ are the indices sampled in line 22 of the algorithm, then using the distribution of U in (32) and the conditional distribution of I^b in (33), we find that the value of β_ℓ is equal to the following,

$$\begin{aligned}\beta_\ell &= s \cdot f_b \cdot \prod_{a=1}^{b} p^a_{h(i_a)} q^a_{i_a} \\ &= s \cdot \Pr[U = b] \cdot \prod_{a=1}^{b} \Pr\left[I_a = i_a | I_1 = i_1, \cdots I_{a-1} = i_{a-1}\right] \\ &= s \cdot \Pr\left[I^b = (i_1, i_2, \dots i_b)\right] \cdot \Pr[U = b],\end{aligned}$$

where p^a and q^a are the distributions computed in lines 18 and 21 of the algorithm. Hence, for any $b = 1, \dots q$ and any $i_1, i_2, \cdots i_b \in [d]$, the distribution of $S_{\ell,\star}$ is,

$$\begin{aligned}&\Pr\left[S_{\ell,\star} = \beta_\ell^{-1/2}\big(\underbrace{0, 0, \dots 0}_{\frac{d^b-1}{d-1} \text{ zeros}}, e_{i_1} \otimes e_{i_2} \otimes \cdots e_{i_b}, \underbrace{0, 0, \dots 0}_{\frac{d^{q+1}-d^{b+1}}{d-1} \text{ zeros}}\big)\right] \\ &\quad = \Pr\left[I^b = (i_1, i_2, \dots i_b)\right] \cdot \Pr[U = b] = \frac{\beta_\ell}{s}. \end{aligned} \tag{34}$$

Furthermore, if $b = 0$ is the degree sampled in line 13 of the algorithm then $\beta_\ell = s \cdot f_0 = s \cdot \Pr[U = 0]$. Thus,

$$\Pr\left[S_{\ell,\star} = \beta_\ell^{-1/2}\big(1, \underbrace{0, 0, \dots 0}_{\frac{d^{q+1}-d}{d-1} \text{ zeros}}\big)\right] = \Pr[U = 0] = \frac{\beta_\ell}{s}.$$

Now to ease the notation we define $Y_k^{(c)} := \bigotimes_{j=c}^{q} S_k^{(j)} X$ for every $k \in [m']$ and $c \in [q]$, where $S_k^{(c)}$ are the SRHT sketches with shared signs drawn in line 4 of the algorithm. Using the definition of $\text{TNORM}^{(k)}$ in line 5 and by invoking Lemma 3.1 we have the following inequality for any $k \in [m']$ and any $j = 0, 1, \dots q$:

$$\text{TNORM}^{(k)}.\text{QUERY}\left(v, q - j\right) \in \left(1 \pm \frac{1}{40q}\right) \left\| \left(Y_k^{(q-j+1)} \otimes M\right) v \right\|_2^2. \tag{35}$$

By union bounding over $(q+1)m'$ events, (35) holds simultaneously for all $j \in \{0, 1, \dots q\}$, and $k \in [m']$, with high probability. From now on we condition on (35). Now, note that $M = H \cdot (B^\top B + \lambda I)^{-1/2}$ for a random Gaussian matrix H with $d' = \Omega(q^2 \log n)$ rows. Therefore, H is a JL-transform. So if we define $A := (B^\top B + \lambda I)^{-1/2}$ for ease of notation, then with probability $1 - \frac{1}{\text{poly}(n)}$, the following holds for any $j \in \{0, 1, \dots q\}, k \in [m']$:

$$\left\| \left(Y_k^{(q-j+1)} \otimes M\right) v \right\|_2^2 \in \left(1 \pm \frac{1}{80q}\right) \left\| Y_k^{(q-j+1)} \cdot \text{diag}(v) A \right\|_F^2.$$

To obtain the above inequality we used the fact that there is a bijective correspondence between entries of vector $\left(Y_k^{(q-j+1)} \otimes M\right) v$ and matrix $Y_k^{(q-j+1)} \cdot \text{diag}(v) M^\top$. Now, using the above inequality along with the definition of matrices $Y_k^{(c)} := \bigotimes_{j=c}^{q} S_k^{(j)} X$ and by invoking Lemma 4.1, because the number of rows of $S_k^{(c)}$'s is $m'' = \Omega(q^3 + q^2 \kappa \log n)$, the following holds with probability at least $9/10$ for any $j \in \{0, 1, \dots q\}, k \in [m']$,

$$\left\| \left(Y_k^{(q-j+1)} \otimes M\right) v \right\|_2^2 \in \left(1 \pm \frac{1}{39q}\right) \left\| X^{\otimes j} \cdot \text{diag}(v) A \right\|_F^2.$$

By plugging the above into (35), we find that with probability at least $9/10$ the following holds,

$$\text{TNORM}^{(k)}.\text{QUERY}\,(v, q-j) \in \left(1 \pm \frac{1}{19q}\right) \left\|X^{\otimes j} \cdot \text{diag}(v) A\right\|_F^2.$$

By taking the median of $m' = \Omega(\log n)$ independent instances of $\text{TNORM}^{(k)}.\text{QUERY}\,(v, q-j)$, the success probability of the above gets boosted. Thus, by a union bound, with probability at least $1 - \frac{1}{\text{poly}(n)}$, the following holds simultaneously for all $j \in \{0, 1, \dots q\}$,

$$\text{MEDIAN}_{k\in[m']} \left\{\text{TNORM}^{(k)}.\text{QUERY}\,(v, q-j)\right\} \in \left(1 \pm \frac{1}{19q}\right) \left\|X^{\otimes j} \cdot \text{diag}(v) A\right\|_F^2.$$

Therefore, using the above along with (32) and definition of f_j in line 11 of the algorithm as well as $A = (B^\top B + \lambda I)^{-1/2}$, with high probability in n, for any $b = 0, 1, \dots q$ we have

$$\begin{aligned}
\Pr[U = b] &= f_b \\
&\geq \left(1 \pm \frac{1}{9q}\right) \frac{\alpha_b^2 \cdot \left\|X^{\otimes b} \cdot \text{diag}(v) A\right\|_F^2}{\sum_{j=0}^q \alpha_j^2 \cdot \left\|X^{\otimes j} \cdot \text{diag}(v) A\right\|_F^2} \\
&= \left(1 \pm \frac{1}{9q}\right) \frac{\alpha_b^2 \cdot \left\|X^{\otimes b} \cdot \text{diag}(v)(B^\top B + \lambda I)^{-1/2}\right\|_F^2}{\left\|\Phi \cdot (B^\top B + \lambda I)^{-1/2}\right\|_F^2},
\end{aligned} \tag{36}$$

where the last line follows from the definition of $\Phi = \bigoplus_{j=0}^q \alpha_j X^{\otimes j} \text{diag}(v)$.

Moreover, suppose that $b \in \{1, 2, \dots q\}$. From the definition of $\text{TNORM}^{(k)}$ in line 5 and by invoking Lemma 3.1 we have the following inequalities for any $r \in [s']$, $k \in [m']$, $i \in [d]$, and any $a = 1, 2, \dots b$, with probability at least $1 - \frac{1}{\text{poly}(n)}$,

$$\text{TNORM}^{(k)}.\text{QUERY}\left(L_{r,k}^a, a+q-b\right) \in \left(1 \pm \frac{1}{40q}\right) \left\|\left(Y_k^{(a+q-b+1)} \otimes M\right) D^a W_{r,k}^\top\right\|_F^2, \tag{37}$$

$$\text{TNORM}^{(k)}.\text{QUERY}\left(D^a X_{i,\star}^\top, a+q-b\right) \in \left(1 \pm \frac{1}{40q}\right) \left\|\left(Y_k^{(a+q-b+1)} \otimes M\right) D^a X_{i,\star}^\top\right\|_2^2 \tag{38}$$

By union bounding over $qds'm'$ events, (37), and (38) hold simultaneously for all $a \in [b]$, $k \in [m']$, $i \in [d]$, and all $r \in [s']$ with high probability. From now on we condition on (37) and (38).

Furthermore, note that $W_{r,k}$ is defined in line 9 as $W_{r,k} = G_r^k \cdot X_{h^{-1}(r),\star}$, where G_r^k is a degree-1 POLYSKETCH with target dimension $n' = C_3 q^2$. By Lemma 2.3, G_r^k approximately preserves the Frobenius norm of any fixed matrix with constant probability. In particular, for every $a \in [b], r \in [s'], k \in [m']$, with probability at least $19/20$:

$$\left\|\left(Y_k^{(a+q-b+1)} \otimes M\right) D^a W_{r,k}^\top\right\|_F^2 \in \left(1 \pm \frac{1}{80q}\right) \left\|Y_k^{(a+q-b+1)} D^a \left(X_{h^{-1}(r),\star} \otimes M\right)^\top\right\|_F^2. \tag{39}$$

To obtain the above inequality we used the fact that there is a bijective correspondence between entries of $\left(Y_k^{(a+q-b+1)} \otimes M\right) D^a X_{h^{-1}(r),\star}^\top$ and $Y_k^{(a+q-b+1)} D^a \left(X_{h^{-1}(r),\star} \otimes M\right)^\top$. Additionally, we use the fact that $M = H \cdot A$ for a JL-transform H. So, with probability $1 - \frac{1}{\text{poly}(n)}$, the following holds for any $a \in [b], r \in [s']$:

$$\left\|Y_k^{(a+q-b+1)} D^a \left(X_{h^{-1}(r),\star} \otimes M\right)^\top\right\|_F^2 \in \left(1 \pm \frac{1}{80q}\right) \left\|Y_k^{(a+q-b+1)} D^a \left(X_{h^{-1}(r),\star} \otimes A\right)^\top\right\|_F^2.$$

By union bounding over qs' events we can conclude that the above inequality holds simultaneously for all $a \in [b], r \in [s']$. From now on we condition on the above inequality holding. By combining this condition with (39) we find that with probability at least $19/20$ the following holds:

$$\left\|\left(Y_k^{(a+q-b+1)} \otimes M\right) D^a W_{r,k}^\top\right\|_F^2 \in \left(1 \pm \frac{1}{39q}\right) \left\|Y_k^{(a+q-b+1)} D^a \left(X_{h^{-1}(r),\star} \otimes A\right)^\top\right\|_F^2. \tag{40}$$

Using the definition of matrices $Y_k^{(c)} := \bigotimes_{j=c}^{q} S_k^{(j)} X$ and by Lemma 4.1, because the number of rows of $S_k^{(c)}$ is $m'' = \Omega(q^3 + q^2 \kappa \log n)$, the following holds with probability at least $19/20$ for any $a \in [b], r \in [s'], k \in [m']$,

$$\left\| Y_k^{(a+q-b+1)} D^a \left(X_{h^{-1}(r),\star} \otimes A \right)^\top \right\|_F^2 \in \left(1 \pm \frac{1}{80q}\right) \left\| X^{\otimes(b-a)} D^a \left(X_{h^{-1}(r),\star} \otimes A \right)^\top \right\|_F^2.$$

By combining the above with (40) and a union bound, plugging the result into (37) we find that with probability at least $9/10$ the following holds,

$$\text{TNORM}^{(k)}.\text{QUERY}\left(L_{r,k}^a, a\right) \in \left(1 \pm \frac{1}{10q}\right) \left\| X^{\otimes(b-a)} \cdot D^a \left(X_{h^{-1}(r),\star} \otimes A \right)^\top \right\|_F^2.$$

By taking the median of $m' = \Omega(\log n)$ independent instances of $\text{TN}^{(k)}.\text{QUERY}\left(L_{r,k}^a, a\right)$, the success probability of the above gets boosted. Thus, by a union bound, with probability at least $1 - \frac{1}{\text{poly}(n)}$ the following holds simultaneously for all $a \in [b]$ and $r \in [s']$,

$$\text{MEDIAN}_{k\in[m']} \left\{ \text{TNORM}^{(k)}.\text{QUERY}\left(L_{r,k}^a, a\right) \right\} \in \left(1 \pm \frac{1}{10q}\right) \left\| X^{\otimes(b-a)} D^a \left(X_{h^{-1}(r),\star} \otimes A \right)^\top \right\|_F^2 \tag{41}$$

Similarly, we can use the fact that there is a bijective correspondence between the entries of $\left(Y_k^{(a+q-b+1)} \otimes M\right) D^a X_{i,\star}^\top$ and $Y_k^{(a+q-b+1)} D^a \text{diag}\left(X_{i,\star}\right) M^\top$ along with $M = H \cdot A$ to conclude that with probability $1 - \frac{1}{\text{poly}(n)}$, the following holds for any $a \in [b], r \in [s'], k \in [m'], i \in [d]$:

$$\left\| \left(Y_k^{(a+q-b+1)} \otimes M\right) D^a X_{i,\star}^\top \right\|_2^2 \in \left(1 \pm \frac{1}{80q}\right) \left\| Y_k^{(a+q-b+1)} D^a \cdot \text{diag}\left(X_{i,\star}\right) A \right\|_F^2 \tag{42}$$

By a union bound over $qs'm'd$ events we can conclude that the above inequality holds simultaneously for all $a \in [b], r \in [s'], k \in [m'], i \in [d]$. From now on we condition on the above inequality holding. Then by using the definition of matrices $Y_k^{(c)} := \bigotimes_{j=c}^{q} S_k^{(j)} X$ and invoking Lemma 4.1, the following holds with probability at least $19/20$ for any $a \in [b], r \in [s'], k \in [m'], i \in [d]$,

$$\left\| Y_k^{(a+q-b+1)} D^a \text{diag}\left(X_{i,\star}\right) A \right\|_F^2 \in \left(1 \pm \frac{1}{80q}\right) \left\| X^{\otimes(b-a)} \cdot D^a \text{diag}\left(X_{i,\star}\right) A \right\|_F^2$$

By combining this with the condition in (42) and (38) we find that with probability at least $19/20$:

$$\text{TNORM}^{(k)}.\text{QUERY}\left(D^a X_{i,\star}^\top, a+q-b\right) \in \left(1 \pm \frac{1}{19q}\right) \left\| X^{\otimes(b-a)} D^a \text{diag}\left(X_{i,\star}\right) A \right\|_F^2$$

By taking the median of $m' = \Omega(\log n)$ independent instances of $\text{TNORM}^{(k)}.\text{QUERY}\left(D^a X_{i,\star}^\top, a+q-b\right)$, the success probability of the above gets boosted. Thus, by applying the median trick and then using a union bound, with probability at least $1 - \frac{1}{\text{poly}(n)}$ the following holds simultaneously for all $a \in [b], i \in [d]$ and $r \in [s']$,

$$\text{MEDIAN}_{k\in[m']} \left\{ \text{TNORM}^{(k)}.\text{QUERY}\left(D^a X_{i,\star}^\top, a+q-b\right) \right\} \in \left(1 \pm \frac{1}{19q}\right) \left\| X^{\otimes(b-a)} D^a \text{diag}\left(X_{i,\star}\right) A \right\|_F^2$$

Plugging the above inequality along with (41) into (33), we conclude that with high probability the following bound holds simultaneously for all $a \in [b]$ and all $i \in [d]$,

$$\Pr[I_a = i | I_1 = i_1, I_2 = i_2, \cdots I_{a-1} = i_{a-1}] \geq \left(1 - \frac{1}{3q}\right) \cdot \frac{\left\| X^{\otimes(b-a)} D^a \cdot \text{diag}\left(X_{i,\star}\right) A \right\|_F^2}{\left\| X^{\otimes(b-a+1)} D^a A \right\|_F^2}. \tag{43}$$

Thus, using the definition of D^a and $A = (B^\top B + \lambda I)^{-1/2}$, for any $b \in \{1, 2, \dots q\}$, we have

$$\begin{aligned}
\Pr\left[I^b = (i_1, i_2, \cdots i_b)\right] &= \prod_{a=1}^{q} \Pr\left[I_a = i_a | I_1 = i_1, \cdots I_{a-1} = i_{a-1}\right] \\
&\geq \prod_{a=1}^{b} \left(1 - \frac{1}{3q}\right) \frac{\left\|X^{\otimes(b-a)} D^a \cdot \operatorname{diag}\left(X_{i_a,\star}\right) A\right\|_F^2}{\left\|X^{\otimes(b-a+1)} D^a A\right\|_F^2} \\
&\geq \frac{1}{2} \cdot \frac{\left\|\mathbf{1}_n^\top \cdot D^b \cdot \operatorname{diag}\left(X_{i_b,\star}\right) A\right\|_F^2}{\left\|X^{\otimes b} D^1 A\right\|_F^2} \\
&= \frac{1}{2} \cdot \frac{\left\|\left[X^{\otimes b} \cdot \operatorname{diag}(v)(B^\top B + \lambda I)^{-1/2}\right]_{(i_1,i_2,\cdots i_b),\star}\right\|_2^2}{\left\|X^{\otimes b} \cdot \operatorname{diag}(v)(B^\top B + \lambda I)^{-1/2}\right\|_F^2}
\end{aligned}$$

This together with (36), shows that for any $b \in \{1, 2 \dots q\}$, with high probability in n,

$$\begin{aligned}
&\Pr\left[S_{\ell,\star} = \beta_\ell^{-1/2}\big(\underbrace{0, 0, \dots 0}_{\frac{d^b-1}{d-1} \text{ zeros}}, e_{i_1} \otimes e_{i_2} \otimes \cdots e_{i_b}, \underbrace{0, 0, \dots 0}_{\frac{d^{q+1}-d^{b+1}}{d-1} \text{ zeros}}\big)\right] \\
&= \Pr\left[I^b = (i_1, i_2, \dots i_b)\right] \cdot \Pr[U = b] \\
&\geq \frac{1}{3} \cdot \frac{\left\|\left[X^{\otimes b} \operatorname{diag}(v)(B^\top B + \lambda I)^{-1/2}\right]_{(i_1,i_2,\cdots i_b),\star}\right\|_2^2}{\left\|X^{\otimes b} \operatorname{diag}(v)(B^\top B + \lambda I)^{-1/2}\right\|_F^2} \cdot \frac{\alpha_b^2 \cdot \left\|X^{\otimes b} \cdot \operatorname{diag}(v)(B^\top B + \lambda I)^{-1/2}\right\|_F^2}{\left\|\Phi \cdot (B^\top B + \lambda I)^{-1/2}\right\|_F^2} \\
&= \frac{1}{3} \cdot \frac{\alpha_b^2 \cdot \left\|\left[X^{\otimes b} \operatorname{diag}(v)(B^\top B + \lambda I)^{-1/2}\right]_{(i_1,i_2,\cdots i_b),\star}\right\|_2^2}{\left\|\Phi \cdot (B^\top B + \lambda I)^{-1/2}\right\|_F^2}.
\end{aligned}$$

The numerator above is exactly equal to the norm of row $(i_1, i_2, \cdots i_b)$ of the b^{th} block of the matrix $\Phi(B^\top B + \lambda I)^{-1/2}$ (note that Φ has $q+1$ blocks and its b^{th} block is $\alpha_b \cdot X^{\otimes b} \operatorname{diag}(v)$). On the other hand if $b = 0$, we have,

$$\Pr\left[S_{\ell,\star} = \beta_\ell^{-1/2}\big(1, \underbrace{0, 0, \dots 0}_{\frac{d^{q+1}-d}{d-1} \text{ zeros}}\big)\right] = \Pr[U = 0] \geq \left(1 \pm \frac{1}{9q}\right) \frac{\alpha_0^2 \cdot \left\|X^{\otimes 0} \cdot \operatorname{diag}(v)(B^\top B + \lambda I)^{-1/2}\right\|_F^2}{\left\|\Phi \cdot (B^\top B + \lambda I)^{-1/2}\right\|_F^2}.$$

The numerator above is exactly equal to the norm of (the sole row of) the 0^{th} block of the matrix $\Phi(B^\top B + \lambda I)^{-1/2}$.

Because $\frac{\beta_\ell}{s}$ is the probability of sampling row $(i_1, i_2, \cdots i_b)$ in the b^{th} block of the matrix $\Phi(B^\top B + \lambda I)^{-1/2}$ or the sole row of the zero-th block, the above inequalities prove that with high probability, matrix S is a rank-s row norm sampler for $\Phi(B^\top B + \lambda I)^{-1/2}$ as in Definition 2.5.

Runtime: The first expensive step of this algorithm is the computation of M in line 3 which takes $O(m^2 n + q^2 mn \log n)$ operations since B has rank at most m. The next expensive computation is the computation of $S_k^{(c)} X$ for $c \in [q]$ and $k \in [m']$ in line 5 of the algorithm. By Lemma 4.1, the total time to compute these sketched matrices is $O\left((q^4 + q^3\kappa)n \log^2 n + nd \log^2 n\right)$. Another expensive step is the construction of the TNORM$^{(k)}$ data-structure in line 5 for $k \in [m']$. By Lemma 3.1, these DS's for $\epsilon = \frac{1}{40q}$ and all $k \in [m']$ can be formed in total time $O\left(q^4 \log^2 q \cdot n \log^2 n + (q^4 + q^3\kappa)n \log^3 n\right)$.

By Lemma 2.3, matrices $W_{r,k}$ for all $r \in [s']$ and $k \in [m']$ in line 9 of the algorithm can be computed in total time $O\left(q^2 s' n \log^2 n + \log n \cdot \operatorname{nnz}(X)\right)$.

The matrix $W_{r,k}$ for every $k \in [m']$, and $r \in [s']$, has size $O(q^2) \times n$. Thus, by Lemma 3.1, computing the distribution $\{p_r^a\}_{r=1}^{s'}$ in line 18 takes time $O\left(q^4 s' \cdot n \log^2 n \log q\right)$ for a fixed $a \in [b]$ and a fixed $\ell \in [s]$. Therefore, the total time to compute this distribution for all a and ℓ is $O\left(q^8 s^2 \cdot n \log^2 n \log q\right)$.

The runtime of computing the distribution $\{q_i^a\}_{i\in h^{-1}(t)}$ in line 21 depends on the sparsity of $X_{h^{-1}(t),\star}$, i.e., $\operatorname{nnz}\left(X_{h^{-1}(t),\star}\right)$. To bound the sparsity of $X_{h^{-1}(t),\star}$, note that, $\operatorname{nnz}\left(X_{h^{-1}(t),\star}\right) = \sum_{i=1}^{d} \mathbb{1}_{\{i\in h^{-1}(t)\}} \cdot \operatorname{nnz}\left(X_{i,\star}\right)$. Since the hash function h is fully independent, by invoking Bernstein's inequality, we find that, for every $t \in [s']$ and $a \in [b]$, with high probability in n, $\operatorname{nnz}\left(X_{h^{-1}(t),\star}\right) = O\left(\left(\operatorname{nnz}\left(X\right)/s' + n\right)\log n\right)$. By union bounding over qs' events, with high probability in n, $\operatorname{nnz}\left(X_{h^{-1}(t),\star}\right) = O\left(\left(\operatorname{nnz}\left(X\right)/s' + n\right)\log n\right)$, simultaneously for all $t \in [s']$ and $a \in [b]$.

Therefore, by Lemma 3.1, the distribution $\{q_i^a\}_{i\in h^{-1}(t)}$ in line 21 of the algorithm can be computed in total time $O\left(q^3 sn\log^4 n\log q + \log^4 n\log q\cdot\operatorname{nnz}\left(X\right)\right)$ for all $a \in [b]$ and all $\ell \in [s]$.

The total runtime of Algorithm 3 is thus $O(m^2 n + q^8 s^2 n\log^2 n\log q + q^3\kappa n\log^3 n + dn\log^4 n)$.

□

Now we are ready to prove the main result, i.e., Theorem 5.2.

Proof of Theorem 5.2: The theorem follows by invoking Lemmas 2.6 and E.1. To find the sampling matrix Π, run Algorithm 1 on Φ with $\mu = s_\lambda$ and for the RowSampler primitive, invoke Algorithm 3. By Lemma E.1, Algorithm 5 outputs a row norm sampler as per Definition 2.5, with probability $1 - \frac{1}{\operatorname{poly}(n)}$. Therefore, since the total number of times Algorithm 5 is invoked by Algorithm 1 is $\log\frac{\|\Phi\|_F^2}{\epsilon\lambda} = O(\log n)$, by a union bound, the preconditions of Lemma 2.6 are satisfied with high probability. Thus, it follows that Π satisfies the following spectral approximation guarantee

$$\frac{\Phi^\top\Phi + \lambda I}{1+\epsilon} \preceq \Phi^\top\Pi^\top\Pi\Phi + \lambda I \preceq \frac{\Phi^\top\Phi + \lambda I}{1-\epsilon}.$$

The only thing that remains is to bound the runtime. In the proof of Lemma 2.6 in (Woodruff & Zandieh, 2020), it is shown that with high probability at any iteration $t \in [T]$ of Algorithm 1, the following holds,

$$\frac{\Phi^\top\Phi + \lambda_t I}{1+\epsilon} \preceq \Phi^\top\Pi_t^\top\Pi_t\Phi + \lambda_t I \preceq \frac{\Phi^\top\Phi + \lambda_t I}{1-\epsilon}.$$

Therefore, $\left\|\Phi^\top\Pi_t^\top\Pi_t\Phi\right\| = O(\|\Phi^\top\Phi\|)$. Now note that Algorithm 1 invokes the RowSampler primitive $T = \log\frac{\|\Phi\|_F^2}{\epsilon\lambda} = O(\log n)$ times. Thus, by Lemma E.1, the runtime of finding Π is the sum of $O\left(\frac{q^8 s_\lambda^2 n\log^4 n}{\epsilon^4} + \sqrt{\frac{\left\|\Phi^\top\Pi_t^\top\Pi_t\Phi\right\|}{\lambda_t}}q^3 n\log^3 n + nd\log^4 n\right)$ for all $t \in [T]$. Since $\lambda_t = 2^{T-t}\lambda$ has a geometric decay, the total time complexity is $O\left(\frac{q^8 s_\lambda^2 n\log^5 n}{\epsilon^4} + \sqrt{\frac{\|\Phi^\top\Phi\|}{\lambda}}q^3 n\log^3 n + nd\log^5 n\right)$.

□

E.1. Application to Gaussian Kernel

In this section we show how to use Theorem 5.2 to spectrally approximate the Gaussian kernel matrix on a dataset with bounded radius. Specifically, we prove Corollary 5.3:

Proof of Corollary 5.3: Our approach is to show that there exists a GPK that tightly approximates the Gaussian kernel matrix and then invoke Theorem 5.2. We start by letting $X \in \mathbb{R}^{d\times n}$ be the matrix whose columns are data-points $x_1, \dots x_n$. Also, let $q = \Theta\left(r + \log\frac{n}{\epsilon\lambda}\right)$ and define $\alpha \in \mathbb{R}^{q+1}$ as $\alpha_j := 1/\sqrt{j!}$ for every $j = 0, 1, \dots q$. Additionally, let $v \in \mathbb{R}^n$ be defined as $v_i := e^{-\|x_i\|_2^2/2}$ for $i \in [n]$. Now we define the GPK kernel matrix $\widetilde{K} \in \mathbb{R}^{n\times n}$ corresponding to the above mentioned q, X, α, and v, i.e., $\widetilde{K} := \operatorname{diag}(v)\left(\sum_{j=0}^{q}\alpha_j^2\cdot X^{\otimes j\top}X^{\otimes j}\right)\operatorname{diag}(v)$. Also let $\widetilde{\Phi}$ be the feature matrix corresponding to $\widetilde{K}$ defined as per (4). Then by invoking Theorem 5.2 we can find a sampling matrix Π in time $O\left(\frac{q^8 s_\lambda^2 n\log^5 n}{\epsilon^4} + \sqrt{\frac{\|K\|}{\lambda}}q^3 n\log^3 n + nd\log^5 n\right) = \widetilde{O}\left(\frac{r^8 s_\lambda^2 n}{\epsilon^4} + \sqrt{\frac{\|K\|}{\lambda}}r^3 n + nd\right)$ such that with high probability in n,

$$\frac{\widetilde{K} + \lambda I}{1+\epsilon/3} \preceq \widetilde{\Phi}^\top\Pi^\top\Pi\widetilde{\Phi} + \lambda I \preceq \frac{\widetilde{K} + \lambda I}{1-\epsilon/3}.$$

Now all that is left to do is to show that

$$\frac{K+\lambda I}{1+\epsilon/3} \preceq \widetilde{K}+\lambda I \preceq \frac{K+\lambda I}{1-\epsilon/3}.$$

To prove the above we note that since K and $\widetilde{K}$ are PSD matrices, it suffices to prove $\left\|\widetilde{K}-K\right\| \leq \frac{\epsilon\lambda}{4}$. The reason we have this bound is,

$$\begin{aligned}
\left\|\widetilde{K}-K\right\|^2 &\leq \left\|\widetilde{K}-K\right\|_F^2 \\
&= \sum_{i,j\in[n]} \left|\widetilde{K}_{i,j}-K_{i,j}\right|^2 \\
&= \sum_{i,j\in[n]} \left|\sum_{\ell=0}^{q}\langle x_i,x_j\rangle^\ell/\ell! - e^{\langle x_i,x_j\rangle}\right|^2 \cdot e^{-\|x_i\|_2^2}\cdot e^{-\|x_j\|_2^2} \\
&\leq \sum_{i,j\in[n]} \left|\sum_{\ell=q+1}^{\infty}\langle x_i,x_j\rangle^\ell/\ell!\right|^2 \\
&\leq \sum_{i,j\in[n]} \left|\sum_{\ell=q+1}^{\infty} r^\ell/\ell!\right|^2 \\
&\leq \sum_{i,j\in[n]} \left|\frac{\epsilon\lambda}{4n}\right|^2 \\
&= \frac{\epsilon^2\lambda^2}{16}.
\end{aligned}$$

This completes the proof and shows that,

$$\frac{K+\lambda I}{1+\epsilon} \preceq \widetilde{\Phi}^\top\Pi^\top\Pi\widetilde{\Phi}+\lambda I \preceq \frac{K+\lambda I}{1-\epsilon}.$$

□

E.2. Application to Neural Tangent Kernel

In this section we show how to use Theorem 5.2 to spectrally approximate the kernel matrix corresponding to the NTK defined in (5) on a dataset with bounded radius. Specifically, we prove Corollary 5.4:

Proof of Corollary 5.4: Our approach is to show that there exists a GPK that tightly approximates the NTK and then invoke Theorem 5.2. We start by letting $X \in \mathbb{R}^{d\times n}$ be the matrix whose columns are normalized data points $\frac{x_1}{\|x_1\|_2},\dots\frac{x_n}{\|x_n\|_2}$. Also let $v\in\mathbb{R}^n$ be defined as the vector of norms $v_i := \|x_i\|_2$ for $i\in[n]$. Additionally, let $q=\Theta\left(\frac{n^2r^2}{\epsilon^2\lambda^2}\right)$ and define the vector of coefficients $\alpha\in\mathbb{R}^{2q+3}$ as follows for every $j=0,1,\dots 2q+2$:

$$\alpha_j := \begin{cases} \frac{1}{\pi} & \text{if } j=0 \\ 1 & \text{if } j=1 \\ 0 & \text{if } j>1 \text{ is odd} \\ \frac{1}{\pi}\cdot\frac{(j+1)\cdot(j-2)!}{2^{j-2}((j/2-1)!)^2\cdot(j-1)\cdot j} & \text{if } j>1 \text{ is even} \end{cases}.$$

Now we define the GPK kernel matrix $\widetilde{K}\in\mathbb{R}^{n\times n}$ corresponding to the abovementioned q, X, α, and v, i.e., $\widetilde{K} := \mathrm{diag}(v)\left(\sum_{j=0}^{q}\alpha_j^2\cdot X^{\otimes j\top}X^{\otimes j}\right)\mathrm{diag}(v)$. Also let $\widetilde{\Phi}$ be the feature matrix corresponding to $\widetilde{K}$ defined as per (4). Then by invoking Theorem 5.2 and also noting that the definition of NTK in (5) implies $\|K\|\leq \mathrm{tr}(K)=2n$, we can find

a sampling matrix Π in time $O\left(\frac{q^8 s_\lambda^2 n \log^5 n}{\epsilon^4} + \sqrt{\frac{\|K\|}{\lambda}} q^3 n \log^3 n + nd \log^5 n\right) = \widetilde{O}\left(\left(\frac{nr}{\epsilon\lambda}\right)^{16} \frac{s_\lambda^2 n}{\epsilon^4} + nd\right)$ such that with high probability in n,

$$\frac{\widetilde{K} + \lambda I}{1+\epsilon/3} \preceq \widetilde{\Phi}^\top \Pi^\top \Pi \widetilde{\Phi} + \lambda I \preceq \frac{\widetilde{K} + \lambda I}{1-\epsilon/3}.$$

Now all that is left to do is to show that

$$\frac{K + \lambda I}{1+\epsilon/3} \preceq \widetilde{K} + \lambda I \preceq \frac{K + \lambda I}{1-\epsilon/3}.$$

To prove the above we note that since K and $\widetilde{K}$ are PSD matrices, it suffices to prove $\left\|\widetilde{K} - K\right\| \leq \frac{\epsilon\lambda}{4}$. To prove this bound note that the Taylor series expansion of function $k_{\texttt{ntk}}(\beta)$ defined in (5) is the following,

$$k_{\texttt{ntk}}(\beta) \equiv \frac{1}{\pi} + \beta + \frac{1}{\pi} \sum_{\ell=0}^{\infty} \frac{(2\ell+3)\cdot(2\ell)!}{2^{2\ell}(\ell!)^2 \cdot (2\ell+1)(2\ell+2)} \cdot \beta^{2\ell+2}.$$

Therefore, we can write

$$\begin{aligned}
\left\|\widetilde{K} - K\right\|^2 &\leq \left\|\widetilde{K} - K\right\|_F^2 \\
&= \sum_{i,j\in[n]} \left|\widetilde{K}_{i,j} - K_{i,j}\right|^2 \\
&= \sum_{i,j\in[n]} \left| \frac{1}{\pi} + \frac{\langle x_i, x_j\rangle}{\|x_i\|\|x_j\|} + \frac{1}{\pi}\sum_{\ell=0}^{q} \frac{(2\ell+3)\cdot(2\ell)!}{2^{2\ell}(\ell!)^2(2\ell+1)(2\ell+2)} \left(\frac{\langle x_i, x_j\rangle}{\|x_i\|\|x_j\|}\right)^{2\ell+2} - k_{\texttt{ntk}}\left(\frac{\langle x_i, x_j\rangle}{\|x_i\|\|x_j\|}\right)\right|^2 \cdot \|x_i\|_2^2 \|x_j\|_2^2 \\
&= \sum_{i,j\in[n]} \left| \frac{1}{\pi}\sum_{\ell=q+1}^{\infty} \frac{(2\ell+3)\cdot(2\ell)!}{2^{2\ell}(\ell!)^2(2\ell+1)(2\ell+2)} \left(\frac{\langle x_i, x_j\rangle}{\|x_i\|\|x_j\|}\right)^{2\ell+2}\right|^2 \cdot \|x_i\|_2^2 \|x_j\|_2^2 \\
&\leq \sum_{i,j\in[n]} \left| \frac{1}{\pi}\sum_{\ell=q+1}^{\infty} \frac{(2\ell+3)\cdot(2\ell)!}{2^{2\ell}(\ell!)^2(2\ell+1)(2\ell+2)}\right|^2 \cdot r^2 \\
&= \frac{n^2 r^2}{\pi^2} \cdot \left| \sum_{\ell=q+1}^{\infty} \frac{(2\ell+3)\cdot(2\ell)!}{2^{2\ell}(\ell!)^2(2\ell+1)(2\ell+2)}\right|^2 \\
&\leq \frac{n^2 r^2}{\pi^2} \cdot \left| \sum_{\ell=q+1}^{\infty} \frac{1}{2\ell^{3/2}}\right|^2 \\
&\leq \frac{n^2 r^2}{4\pi^2 q} \leq \frac{\epsilon^2\lambda^2}{16}.
\end{aligned}$$

This completes the proof and shows that,

$$\frac{K + \lambda I}{1+\epsilon} \preceq \widetilde{\Phi}^\top \Pi^\top \Pi \widetilde{\Phi} + \lambda I \preceq \frac{K + \lambda I}{1-\epsilon}.$$

□