6

Vector, Matrix and Linear Algebraic Summaries

In this chapter, we show how ideas from the summaries described previously can be applied directly or adapted to solve problems that arise in the context of linear algebra: computations on vectors, matrices, and their generalizations to tensors.

For vectors, many of the summaries apply directly. We can often think of a vector as being synonymous with a frequency distribution, where the ith entry in the vector corresponds to the frequency of an element i. Queries for a particular element of a vector then correspond to a point query in the frequency distribution. Frequency moments have a natural interpretation in terms of vector norms. However, there are some problems over vectors which do not correspond to a natural question over frequency distributions, and so require some additional work.

In the case of matrices, many of the ideas and techniques from vector summaries carry over — in the extreme case, we can think of matrices as ordered collections of vectors. We typically describe results in terms of the Frobenius norm $\|M\|_F$ of a matrix M, an "entry-wise" norm which measures the (square root of the) sum of squares of entries, i.e., $\|M\|_F = \sqrt{\sum_{i,j} M_{i,j}^2}$.

6.1 Vector Computations: Euclidean Norm and Inner Product Estimation

Given a high-dimensional vector v, we would like to find its Euclidean norm, $\|v\|_2$. This problem is by now very well understood, and several techniques directly solve it: the AMS Sketch (Section 3.6) in particular addresses this problem, but more generally, methods that implement the Johnson-Lindenstrauss transform by computing an appropriate "sketch" of the vector can be applied. These find an ϵ-relative error approximation of $\|v\|_2$.

Equipped with such an approximation, we can solve a number of other problems on vectors. Most immediately, given vectors v and w we can find the Euclidean distance between them, defined as $\|v-w\|_2$. Functionally, this can be achieved by building a summary of v, and a summary of the vector $(-w)$ and applying a MERGE operation to these two. Because the summaries are linear transformations of the input vectors, one can more directly think of the summary of v as $sk(v)$, where $sk(\cdot)$ denotes the operation of building the (sketch) summary. Then $sk(v-w) = sk(v) - sk(w)$, and so we can obtain the sketch of the difference by taking the difference of the sketches. The Euclidean distance can therefore be estimated with relative error.

We can also use this property to find an estimate of the inner product between two vectors, albeit with additive error. The inner product between u and v is given by the sum of products of corresponding indices, defined as

$$u \cdot v = \sum_i u_i v_i$$

This can also be written in terms of Euclidean norm, using some algebra. Observe that

$$\|u+v\|_2^2 = \|u\|_2^2 + \|v\|_2^2 + 2(u \cdot v).$$

To see this, note that the contribution to the sum represented by the left hand side from any index i is $u_i^2 + v_i^2 + 2u_i v_i$. Therefore, we can write $(u \cdot v) = \frac{1}{2}(\|u+v\|_2^2 - \|u\|_2^2 - \|v\|_2^2)$. Using sketches, we can estimate the right hand side as our approximation to the inner product.

This estimate does *not* provide overall relative error. Each of the component quantities is found with relative error, but the additions and subtractions mean that the error stays of the magnitude of $\|u\|_2^2 + \|v\|_2^2$ (at least, this gives a quick bound). It is reasonable to ask whether a relative error can be obtained. In particular, there are many applications where a relative error guarantee for inner product would be most helpful. However, there are strong lower bounds (see Section 10.3) showing that such a result cannot be obtained in general. In essence, this is because such a sketch would allow us to distinguish between vectors which have inner product zero (i.e., are orthogonal), and which have inner product very close to zero (almost orthogonal).

A more careful use of summaries can obtain an improved additive error bound. Suppose, for the sake of concreteness, that we are using the AMS Sketch to summarize u and v. Each summary is formed as an array of $d \times t$ counters. A QUERY operation for the inner product takes the inner product of each of the d rows of the pair of summaries, then finds the median of these. Then a similar analysis to the original use of the sketches shows that the result gives an estimate of $u \cdot v$ with additive error proportional to $\frac{1}{\sqrt{t}} \cdot \|u\|_2 \|v\|_2$, with probability $1 - \exp(-d)$. With the usual setting of $t = O(1/\epsilon^2)$ and $d = O(\log 1/\delta)$, the result gives error $\epsilon \|u\|_2 \|v\|_2$ with probability $1 - \delta$.

Further Discussion. We briefly sketch the outline of the proof, which can be viewed as a generalization of the proof for the estimator for the ℓ_2 norm. Recall the construction of the AMS Sketch, which is defined based on hash functions g and h, which map to arrays of counters C. As before, define X_j to be the estimate of $u \cdot v$ obtained from the j'th row of the sketch, i.e., $X_j = \sum_{k=1}^{t} C_u[j,k] C_v[j,k]$. It follows quickly that

$$\begin{aligned} \mathsf{E}[X_j] &= \sum_{k=1}^{t} \mathsf{E}[C_u[j,k] C_v[j,k]] \\ &= \sum_{k=1}^{t} \sum_{h_j(i)=k} u_i v_i + \sum_{i \neq \ell, h_j(i) = h_j(\ell) = k} \mathsf{E}[g_j(i)] \mathsf{E}[g_j(\ell)] u_i v_\ell . \end{aligned}$$

The first term in this sum is $u \cdot v$, and the second term is zero, due to the random properties of the g_j hash functions.

Similarly, for the variance, after expanding out and collapsing the terms that are zero in expectation, we obtain that

$$\mathsf{Var}[X_j] \leq 4 \|u\|_2^2 \|v\|_2^2 / t$$

This bound on the variance ensures that the error is at most $\epsilon \|u\|_2 \|v\|_2$ with constant probability when we set the parameter t proportional to $1/\epsilon^2$. This error probability is reduced to δ when we choose $d = O(\log 1/\delta)$ and take medians, following the Chernoff bounds argument (Section 1.4).

History and Background. The use of the summary to estimate the inner product of vectors was described in a follow-up work by Alon, Matias, Gibbons and Szegedy [9], and the analysis was similarly generalized to the fast version of the AMS Sketch by Cormode and Garofalakis [62].

6.2 ℓ_p norms and Frequency Moments

Recall that, given a vector v, its ℓ_p norm is $\|v\|_p = (\sum_i v_i^p)^{1/p}$. For vectors formed as the frequency distribution of a series of updates, the k'th frequency moment is defined as $F_k(v) = \sum_i v_i^k$. These definitions are sufficiently similar (up to taking the p'th root) that we will use them interchangeably. In earlier sections, we have discussed summaries that allow estimating the ℓ_2 norm (AMS Sketch), and other ℓ_p norms for $p < 2$ (ℓ_p sketch). We now consider how to estimate the ℓ_p norm for $p > 2$.

An elegant solution is provided via the technique of ℓ_p sampling. We provide an outline here, since the details are rather more technical, and so this approach may be primarily of theoretical interest. In essence, the idea is to sample each element of vector v with probability proportional to its contribution to the (squared) ℓ_2 norm, i.e., apply an ℓ_2-sampling sketch. Then for each sampled element i, include a contribution as a function of its weight v_i so that in expectation its contribution is v_i^p. Summing expectations over all elements, we will obtain an estimate of $\|v\|_p^p$. The variance of this quantity can be bounded, and so the usual approach of taking sufficient repetitions via averaging and median can be used to provide an estimate with accuracy guarantees. The number of repetitions required is $O(n^{1-2/p}\epsilon^{-2})$, where n represents the dimensionality of the vector v. This polynomial dependence on n for $p > 2$ is known to be optimal, and imposes a large space cost ($O(n^{1/3})$ for $p = 3$, for example), which is further compounded since each repetition of the ℓ_2 sampling requires multiple sketches to be kept. Consequently, this approach seems to need substantial engineering and careful parameter setting in order to be put into practice.

Further Discussion. We provide more details of the analysis, to give more insight into the method. Full details are provided in the references listed below.

Suppose we could sample perfectly according to the ℓ_2 distribution. Then the chance of picking index i from vector v is exactly

$v_i^2/\|v\|_2^2$. Further, assume that our ℓ_p sampling summary allows us to estimate v_i and $\|v\|_2$ accurately. Then our estimator when we sample i is $X = v_i^{p-2}\|v\|_2^2$. Observe that this is correct in expectation:

$$\mathsf{E}[X] = \sum_i \Pr[i \text{ is sampled}] \cdot v_i^{p-2} = \sum_i \frac{v_i^2}{\|v\|_2^2} v_i^{p-2}\|v\|_2^2 = \sum_i v_i^p = \|v\|_p^p$$

The variance of this estimator can be computed similarly, as

$$\mathsf{Var}[X] \le \mathsf{E}[X^2] = \sum_i \frac{v_i^2}{\|v\|_2^2}(v_i^{p-2}\|v\|_2^2)^2 = \sum_i v_i^2 v_i^{2p-4}\|v\|_2^2 = \|v\|_{2p-2}^{2p-2}\|v\|_2^2$$

In order to provide a more useful bound, we need to rewrite this last quantity in terms of $\mathsf{E}[X]^2$. First, for any $p \ge 2$, we have that $\|v\|_{2p-2} \le \|v\|_p$. Via Hölder's inequality, we can show that $\|v\|_2^2 \le n^{(p-2)/p}\|v\|_p^2$. Combining these two facts, we obtain that

$$\|v\|_{2p-2}^{2p-2}\|v\|_2^2 \le n^{1-2/p}\|v\|_p^{2p}$$

That is, we have that $\mathsf{Var}[X] \le n^{1-2/p}\mathsf{E}[X]^2$. Taking the average of $n^{1-2/p}/\epsilon^2$ repetitions reduces the variance by a corresponding factor. This allows us to apply the Chebyshev inequality to show that this mean is with an $1+\epsilon$ factor of $\mathsf{E}[X]$ with constant probability. Repeating $O(\log 1/\delta)$ times and taking the median amplifies this to probability $1-\delta$, via the Chernoff bounds argument.

In order to put this into practice, we need to remove the idealized sampling assumption. That is, rather than sampling with probability exactly $v_i^2/\|v\|_2^2$, we use an ℓ_p sampler to sample with approximately this probability. Similarly, we can only guarantee to find approximations of v_i and $\|v\|_2$. Putting these approximations into the analysis complicates the notation, but otherwise does not change the structure of the argument. Some care is needed in setting the accuracy parameters of the sampling and sketching in order to obtain an overall ϵ guarantee, but otherwise this is a straightforward exercise to analyze the algorithm's properties.

History and Background. The idea of using ℓ_p sampling to estimate ℓ_p norms and frequency moments is due to Coppersmith and Kumar [58], but it was not until Monemizadeh and Woodruff provided the first ℓ_2 sampling algorithm that this was possible [181]. A more detailed survey of ℓ_p sampling and its applications is given by Cormode and

Jowhari [66]. The presentation here follows outlines due to Andrew McGregor presented in slides on "sketches for ℓ_p sampling".

6.3 Full matrix multiplication

Perhaps the most fundamental problem in linear algebra is to perform matrix multiplication: given matrices A and B, compute their matrix product AB. Naive computation of the product for square $n \times n$ matrices takes time proportional to n^3, but faster computation is possible. Strassen's algorithm rewrites the computation in terms of a smaller number of terms to obtain $O(n^{2.807})$. Theoretical algorithms exist which have cost $O(n^{2.372})$, although these are considered far from practical. These approaches work with the full matrices, and provide exact solutions. However, thanks to summaries, it is possible to provide approximate answers.

A starting point is to observe that each entry of the matrix product is an inner product: the entry $C_{i,j}$ is given by $A^{(i)} \cdot B_{(j)}$, where $A^{(i)}$ denotes the i'th row of matrix A, and $B_{(j)}$ the j'th column of B. We can then use sketches (such as AMS Sketch) to approximate these quantities. Assume we sketch every row of A and sketch every column of B. Then we can estimate (with constant probability of success) any desired entry of C, with error proportional to $\epsilon\|A^{(i)}\|_2\|B_{(j)}\|_2$, for sketches of size $O(1/\epsilon^2)$. The space required is $O(n/\epsilon^2)$ to store the n sketches of each row of A and each column of B.

We can therefore take this approach to estimate every entry of C, as approximate matrix $\hat{C}$, using space $O(n/\epsilon^2)$ and taking time $O(n^2/\epsilon^2)$. The squared error in each entry is $\epsilon^2\|A^{(i)}\|_2^2\|B_{(j)}\|_2^2$. This may appear somewhat large, but if we sum this over all entries to get the total squared error, we obtain

$$\|(\hat{C} - AB)\|_F^2 \leq \sum_{i,j} \epsilon^2\|A^{(i)}\|_2^2\|B_{(j)}\|_2^2 = \epsilon^2(\sum_i \|A^{(i)}\|_2^2)(\sum_j \|B_{(j)}\|_2^2) = \epsilon^2\|A\|_F^2\|B\|_F^2$$

where $\|\cdot\|_F$ denotes the Frobenius norm as defined at the start of this chapter. Then we have that

$$\|(\hat{C} - AB)\|_F \leq \epsilon\|A\|_F\|B\|_F$$

Further Discussion. Note that to have this property hold, we require each entry to meet its guarantee. This is achievable by increasing the size of the sketches by a factor of $O(\log n)$. Then each entry is estimated accurately with probability $1 - 1/n^3$, and so all n^2 entries are accurate with probability $1 - 1/n$, by a union bound.

Intuitively, this increased size of the sketches does not seem necessary: while some entries might exceed their permitted error bounds, others might lie comfortably within them, and we might expect this variation to cancel out on average. Essentially this is indeed the case, although proving it is rather more fiddly.

A first attempt is to consider the total squared error as a random variable, and apply error bounds to this, rather than the each individual entry-wise error. We have already calculated the expectation and variance of the entry-wise errors, so we can immediately obtain expectation and variance of their sum. It might seem that we can then immediately apply the Chebyshev and Chernoff argument to this. However, there is a twist: previously, we were able to take the median of a (scalar) quantity to obtain an estimate with high probability of falling within our desired error bounds. But now, each sketch yields a candidate approximate matrix product: how should we take the median of a collection of matrices?

One approach is to use the fact that we can accurately estimate the norms of matrices, and the norm of differences between pairs of matrices. We can argue that with probability at least $1 - \delta$, one of our matrices is "central" among the estimates built from sketches: it is at most a distance of $\|A\|_F\|B\|_F/2$ to each of at least half of the other estimated matrices. This follows by arguing that there is a constant probability that each estimated matrix is "good"—that is, close (as a function of $\|A\|_F\|B\|_F$) to the target C; and therefore, by the triangle inequality, any pair of "good" matrices must also be close to each other. Performing this distance test helps to identify one good matrix, which can be reported as the approximate product.

A more direct approach is to go back to the original approach and argue that the sketches provide a stronger property: that they estimate not just one inner product accurately with constant probability, but that (with constant probability) they estimate *all* inner products accurately, and so we can work directly with the results of sketching. This stronger property is referred to as *subspace embedding*, and is discussed in more detail when we consider the regression problem.

History and Background. The observation that (dense) sketches allow us to estimate all entries is due to Sarlós [198], and uses an additional randomized trick to pick the estimate of C that is closest to C. Clarkson and Woodruff argue that sparse sketches are sufficient, and introduce the technique based on finding one matrix that is close to a majority of the others [51]. They also show a stronger bound on sketch estimators, when constructed with more powerful hash functions. A more thorough survey which expands on the subspace embedding approach is given by Woodruff [228].

6.4 Compressed matrix multiplication

Given two matrices A and B, both of size $n \times n$, computing their product $C = AB$ is a fundamental problem. Despite years of extensive research, exact matrix multiplication remains an expensive operation. In this section, we present a method that allows us to compute C approximately in a compressed form. More precisely, we first compute the Count Sketch of each column of A and each row of B. Then from these sketches we will compute the Count Sketch of C, interpreted as a long vector of size n^2. As a result, all the properties of the Count Sketch from Section 3.5 apply here. For example, from this sketch, we can estimate any entry in C with error at most $\|C\|_F/\sqrt{t}$ with high probability, where, as before, $\|\cdot\|_F$ denotes the *Frobenius norm*, i.e., $\|C\|_F = \sqrt{\sum_{i,j} C_{i,j}^2}$.

Recall from Section 3.5 that (one row of) the Count Sketch of a vector v of size n is another vector $S(v)$ of size $t \ll n$ (both vectors are indexed starting from 0), where

$$S(v)[k] = \sum_{i \in [n]: h(i)=k} g(i) v_i, k = 0, \ldots, t-1.$$

Here, $h : [n] \to [t]$ is a pairwise independent hash function, and $g : [n] \to \{-1, +1\}$ is also a pairwise independent hash function. Also recall that $S(v)$ can be constructed from v easily in one pass; in particular, this is still true if v is represented in a "compressed form", i.e., a list of (index, value) pairs consisting of all the nonzero entries of the vector.

Let u, v be two column vectors of size n. At the heart of the method is a nice way to compute $S(uv^T)$ from $S(u)$ and $S(v)$, where uv^T is the *outer product* of u and v, and $S(uv^T)$ is a Count Sketch built on uv^T interpreted as a vector of size n^2. Recall that the outer product of u and v is

$$uv^T = \begin{bmatrix} u_1v_1 & u_1v_2 & \dots & u_1v_n \\ u_2v_1 & u_2v_2 & \dots & u_2v_n \\ \vdots & \vdots & \ddots & \vdots \\ u_nv_1 & u_nv_2 & \dots & u_nv_n \end{bmatrix}.$$

Let the two hash functions used in constructing $S(u)$ and $S(v)$ be h_u, g_u and h_v, g_v, respectively. The idea is to compute the *convolution* of $S(u)$ and $S(v)$. Recall that given two vectors a and b of size n, their convolution $a * b$ is vector of size $2n - 1$, where

$$(a * b)[k] = \sum_{j=0}^{k} a[j]b[k-j], k = 0, \dots, 2n-2.$$

The nice thing about convolution is that it can be computed efficiently using the Fast Fourier Transform (FFT) in $O(n \log n)$ time [59].

Next, we will interpret the convolution $S(u) * S(v)$ as a sketch of a new object. The convolution yields a vector of size $2t - 1$, where for $k = 0, 1, \dots, 2t-2$,

$$\begin{aligned}(S(u) * S(v))[k] &= \sum_{j=0}^{k} S(u)[j]S(v)[k-j] \\ &= \sum_{j=0}^{k}\left(\sum_{i\in[n]:h_u(i)=j} g_u(i)u_i\right)\left(\sum_{i\in[n]:h_v(i)=k-j} g_v(i)v_i\right) \\ &= \sum_{i,j\in[n]:h_u(i)+h_v(j)=k} g_u(i)g_v(j)u_iv_j.\end{aligned}$$

This is exactly a Count Sketch built on uv^T, but using hash functions $h(i, j) = h_u(i) + h_v(j)$ and $g(i, j) = g_u(i)g_v(j)$. Since both g_u and g_v are pairwise independent functions mapping $[n]$ to $\{-1, +1\}$, $g(i, j) = g_u(i)g_v(j)$ is still a pairwise independent function mapping $[n]$ to $\{-1, +1\}$. However, $h(i, j) = h_u(i) + h_v(j)$ is no longer a pairwise independent function. In particular, it is not uniform on $[2t-1]$. The last trick is to fold $S(u) * S(v)$ into a vector of size t, getting a true Count Sketch on uv^T, where

$$S(uv^T)[k] = \begin{cases} (S(u) * S(v))[k] + (S(u) * S(v))[k+t], & k = 0, 1, \dots, t-2; \\ (S(u) * S(v))[k], & k = t-1. \end{cases}$$

$$= \sum_{i \in [n]: h_u(i)+h_v(j) \bmod t = k} g_u(i) g_v(j) u_i v_j.$$

Observing that $h(i, j) = h_u(i)+h_v(j) \bmod t$ is a pairwise independent function from $[n]$ to $[t]$, we conclude that $S(uv^T)$ is a valid Count Sketch on uv^T.

Finally, to use this technique to compute a Count Sketch on $C = AB$, we write A in terms of column vectors $a_1, \ldots, a_n$ and B in terms of row vectors $b_1, \ldots, b_n$. The matrix product AB can be written as a sum of n products of these vectors (effectively, these are outer products), as

$$AB = \sum_{i=1}^{n} a_i b_i^T,$$

and the fact that the Count Sketch is a linear sketch, we can compute the Count Sketch of $C = AB$ as

$$S(C) = \sum_{i=1}^{n} S(a_i b_i^T).$$

Further Discussion. In practice, one does not have to compute the convolution $S(u) * S(v)$ in full and then fold the result as described above. The Count Sketch $S(uv^T)$ can be computed more directly by slightly adjusting the FFT algorithm. In computing the full convolution, we use the *convolution theorem*:

$$S(u) * S(v) = \mathrm{FFT}_{2t}^{-1}(\mathrm{FFT}_{2t}(S(u)) \cdot \mathrm{FFT}_{2t}(S(v))).$$

Here, we perform the FFT using the $(2t)$-th root of unity. However, if we simply use the t-th root of unity ω_t, then the result of the convolution will be the folded $S(uv^T)$ directly. This is also known as a circular convolution. The intuition is that since $\omega_t^t = 1$, it has exactly the effect of doing the $\bmod t$ operation. More detailed arguments can be found in [192].

The algorithm above also easily extends to non-square matrices. Suppose A is an $n_1 \times n_2$ matrix and B is an $n_2 \times n_3$ matrix, then $C = AB$ will be a sum of n_2 outer products. Although each outer product is between two vectors of different lengths, this does not affect the algorithm in any way, as the sketch sizes of the two vectors are always the same, which is t.

The running time of the algorithm is $O(N + n_2 t \log t)$, where N is the number of nonzero entries in A and B. Note that this can be even smaller than $O(n_1 n_3)$, which is the size of C. This is because we compute $C = AB$ in a compressed form, represented by the Count Sketch of C. Reconstructing the full C (approximately) still takes $O(n_1 n_3)$ time, since we have to query the Count Sketch for every entry of C. In [192], a method is given to extract the most significant entries of C more quickly.

History and Background. The algorithm described in this section is due to Pagh [192]. Subsequent work generalized the approach of taking convolutions of sketches to quickly build a sketch of the tensor product of a vector: essentially, a polynomial generalization of the outer product [195].

6.5 Frequent directions

Other summaries of matrices are possible, depending on what properties of the matrix we would like to (approximately) preserve. The FrequentDirections summary captures information about the action of the summarized matrix on other matrices and vectors. It does so by iteratively using the Singular Value Decomposition (SVD) on summaries computed so far to capture the information needed while keeping the summary small.

The summary applies to a data matrix A of dimension $n \times d$, where we assume that n will grow very large, while d is of moderate size. The matrix A is formed as the collection of n row vectors, each of dimension d, where each row vector is seen together, i.e., rows arrive one by one. The summary of A will be a smaller matrix B with up to ℓ rows of dimension d, so that B approximates A. We will focus on the action of A on approximating an arbitrary vector x of dimension d. That is, our objective is to show that our summary B of matrix A ensures that the norm of Bx is close to the norm of Ax.

Operations on the summary. To INITIALIZE a new FrequentDirections summary B, we simply instantiate a new zero matrix of size $\ell \times d$. The UPDATE procedure for a new row vector a_i involves the computation of the SVD of B. We append the new row a_i to B to obtain B', and compute its SVD. This gives us the decomposition $U\Sigma V^T = B'$, so that $U^T U = I_\ell$ and $V^T V = I_d$, i.e., these (singular) vectors are orthonormal. The matrix

Σ is diagonal, consisting of the singular values $\sigma_1 \dots \sigma_\ell$, sorted by magnitude (i.e., $|\sigma_1| \geq |\sigma_2| \geq \dots \geq |\sigma_\ell|$). We use these values to rescale the different "directions" in B, and to "forget" the least important direction. We define a new diagonal matrix Σ' as

$$\Sigma' = \text{diag}\left(\sqrt{\sigma_1^2 - \sigma_\ell^2}, \sqrt{\sigma_2^2 - \sigma_\ell^2} \dots \sqrt{\sigma_\ell^2 - \sigma_\ell^2}\right).$$

Equivalently, we can write $\Sigma' = \sqrt{\Sigma - \sigma_\ell^2 I_\ell}$. This ensures that $\Sigma'_{\ell,\ell} = 0$. We use Σ' as a new set of singular values, and obtain an updated version of the summary B as $B = \Sigma' V^T$. Observe that since $\Sigma_\ell = 0$, the new B has at most $\ell - 1$ non-zero rows, so we drop row ℓ (which is all zeros).

The MERGE procedure builds on the approach of the UPDATE procedure. Given two summary matrices B and C, both of size $\ell \times d$, we can simply take each row of C in turn and UPDATE B with it. This is rather slow (requiring ℓ invocations of the SVD procedure), but we can observe that this is equivalent to the following method. We begin by appending B and C to get a new matrix D of size $2\ell \times d$. We then compute the SVD of D and obtain the corresponding $U\Sigma V^T$. We now compute Σ' based on σ_ℓ^2, noting that σ_ℓ is the ℓ'th largest of the 2ℓ singular values in Σ. Then

$$\Sigma' = \text{diag}\left(\sqrt{\max(\sigma_1^2 - \sigma_\ell^2, 0)}, \sqrt{\max(\sigma_2^2 - \sigma_\ell^2, 0)} \dots \sqrt{\max(\sigma_{2\ell}^2 - \sigma_\ell^2, 0)}\right).$$

Observe that the first ℓ values of Σ' will be non-negative, while the last $\ell + 1$ values are guaranteed to be zero. Then, as before, we compute the new summary as $B = \Sigma' V^T$, and keep only the (at most $\ell - 1$) non-zero rows.

Further Discussion. Recall that we want to show that the final summary we obtain B is similar to the input matrix A in its action on a vector x. In particular, we will consider the quantity $\|Ax\|_2^2 - \|Bx\|_2^2$. We can analyze the accuracy of the summary by considering the effect on the error of answering a particular query. Without loss of generality, let x be a vector of dimension d, with $\|x\|_2 = 1$. We will measure the error in terms of Euclidean norms (of vectors) and Frobenius norms (of matrices). Consider the quantity Ax, which we will approximate by Bx for the final summary B. We want to show that the difference in magnitude of these quantities is not too large, i.e., $\|Ax\|_2^2 - \|Bx\|_2^2$ is bounded. For simplicity, we will focus on the effect of a sequence of UPDATE operations.

A bound on this quantity follows by summing over the error introduced in each subsequent UPDATE operation. We first show that the error is non-negative, i.e., that $\|Ax\|_2^2 - \|Bx\|_2^2 \geq 0$. We'll write $B_{[i]}$ for the summary obtained after the i'th UPDATE row operation for row a_i. We also write $C_{[i]}$ to denote the value of $\sigma_{[i]} V_{[i]}^T$ computed from the SVD in step i. Finally, δ_i is the value of σ_ℓ^2 that is subtracted from the other singular values in the ith UPDATE operation.

Then we can expand

$$\begin{aligned}\|Ax\|_2^2 - \|B_{[n]}x\|_2^2 &= \sum_{i=1}^{n} \left((a_i \cdot x)^2 + \|B_{[i-1]}\|_2^2 x - \|B_{[i]}x\|_2^2 \right) \\ &= \sum_{i=1}^{n} \left(\|C_{[i]}x\|_2^2 - \|B_{[i]}x\|_2^2 \right) \geq 0\end{aligned}$$

This makes use of the fact that $C_{[i]}$ preserves all the information about $B_{[i-1]}$ and a, so that $\|B_{[i-1]}x\|_2^2 + (a_i \cdot x)^2 = \|C_{[i]}x\|_2^2$. Meanwhile, $B_{[i]}$ loses information from $C_{[i]}$, so we have $\|C_{[i]}x\|_2^2 - \|B_{[i]}x\|_2^2 \geq 0$.

Next, we consider placing an upper bound on the error $\|Ax\|_2^2 - \|Bx\|_2^2$. For step i, let $v_j[i]$ denote the j'th column of $V_{[i]}^T$. Then $C_{[i]}x = \sum_{j=1}^{\ell} \sigma_j v_j \cdot x$ and:

$$\begin{aligned}\|C_{[i]}x\|_2^2 &= \sum_{j=1}^{\ell} \sigma_j^2 (v_j \cdot x)^2 \\ &= \sum_{j=1}^{\ell} (\sigma'^2_j + \delta_i)(v_j \cdot x)^2 \\ &= \sum_{j=1}^{\ell} \sigma'^2_j (v_j \cdot x)^2 + \delta_i \sum_{j=1}^{\ell} (v_j \cdot x)^2 \\ &= \|B_{[i]}x\|_2^2 + \delta_i\end{aligned}$$

The final line substitutes the definition of $B_{[i]}$ using the reweighted values in Σ', and the fact that V^T is a unitary matrix while $\|x\|_2 = 1$. We can then sum this over every UPDATE step:

$$\|Ax\|_2^2 - \|B_{[n]}x\|_2^2 = \sum_{i=1}^{n} \left((a_i \cdot x)^2 + \|B_{[i-1]}x\|_2^2 - \|B_{[i]}x\|_2^2 \right)$$

$$= \sum_{i=1}^{n} \left(\|C_{[i]}x\|_2^2 - \|B_{[i]}x\|_2^2 \right)$$
$$\leq \sum_{i=1}^{n} \delta_i \qquad (6.1)$$

To bound δ_i, note that the squared Frobenius norm of a matrix is given by the sum of the squares of its singular values. Hence,

$$\|C_{[i]}\|_F^2 \geq \|B_{[i]}\|_F^2 + \ell\delta_i.$$

In each update, we start by appending the new row a_i, which causes the squared Frobenius norm to increase:

$$\|C_{[i]}\|_F^2 = \|B_{[i-1]}\|_F^2 + \|a_i\|_2^2.$$

Last, $\|A\|_F^2 = \sum_{i=1}^{n} \|a_i\|_2^2$. Rearranging and combining these three results, we obtain that $\|A\|_F^2 \geq \|B_{[n]}\|_F^2 + \ell \sum_{i=1}^{n} \delta_i$. Combining this with (6.1) above, we have our desired bound on the error,

$$0 \leq \|Ax\|_2^2 - \|B_{[n]}x\|_2^2 \leq (\|A\|_F^2 - \|B\|_F^2)/\ell \leq \|A\|_F^2/\ell$$

Via a slightly longer argument, we can show stronger bounds in terms of A_k, which is the optimal k-rank approximation of A. A_k can be found via the SVD of $A = U\Sigma V^T$: defining U_k and V_k as the first k columns of U and V respectively, and Σ_k as the diagonal matrix containing k largest entries of Σ, we have $A_k = U_k\Sigma_k V_k^T$. Then we find

$$0 \leq \|Ax\|_2^2 - \|B_{[n]}x\|_2^2 \leq \|A - A_k\|_F^2/(\ell - k)$$

That is, provided ℓ is chosen suitably larger than k, we obtain a good approximation in terms of A_k.

Properties of the MERGE operation follow similarly, by considering the accuracy of two summaries, and arguing that the total error is bounded in terms of the sum of the local errors.

Implementation Issues. Computing the SVD of an $\ell \times d$ matrix (the central step in the UPDATE operation) takes time $O(d\ell^2)$. This is quite time consuming if carried out every step. The cost can be reduced by reducing the frequency with which SVD is found, but storing some extra space. Essentially, we keep a buffer of the most recent ℓ rows, then perform a MERGE operation with the running summary. This amortizes the cost of the SVD to $O(d\ell)$ time per update, and keeps the space used to $O(d\ell)$ also.

History and Background. The FrequentDirections summary was introduced by Liberty [165], taking inspiration from the behavior of frequent items algorithms, particularly the MG summary. Additional results, including lower bounds and the extension to handling a MERGE operation are due to Woodruff [227] and Ghashami and Phillips [107] respectively. Our presentation follows the outline of Ghashami et al. [106]. Due to its simplicity and flexibility, the summary has been widely used to summarize large data sets for machine learning, such as in feature selection, dimensionality reduction and outlier detection.

Available Implementations. The FrequentDirections summary has garnered a lot of interest since it was first described, and several implementations exist. A Python implementation due to Liberty and Ghashami is available, `https://github.com/edoliberty/frequent-directions`. A Java implementation is available through the emerging DataSketches vector library, `https://github.com/apache/incubator-datasketches-vector`.

6.6 Regression and Subspace Embeddings

The problem of regression can be expressed in terms of matrices and vectors. Given a "data" matrix A of size $n \times d$, and a corresponding "response" vector b of dimension n, the aim is to find a vector x of coefficients so as to minimize the quantity $\|Ax - b\|_p$. That is, our goal is to find

$$\arg\min_x \|Ax - b\|_p$$

The norm p to minimize is a parameter of the problem. We will focus on the common case of $p = 2$, which gives the Ordinary Least Squares regression problem.

We make use of a useful concept, an (oblivious) subspace embedding. Given an $n \times d$ matrix A, the matrix S is a subspace embedding for A if, for all $x \in \mathcal{R}^d$,

$$(1 - \epsilon)\|Ax\|_2^2 \leq \|SAx\|_2^2 \leq (1 + \epsilon)\|Ax\|_2^2. \qquad (6.2)$$

In other words, (SA) approximates A in terms of the length of vectors x under the action of the matrix. We say that the embedding is *oblivious* if S is sampled at random from a distribution of matrices, independent of A. That is, we do not have to look at A to be (almost) certain that a sampled S gives a subspace embedding for A.

Observe that this definition appears quite similar to that for the Johnson-Lindenstrauss transform Sparse JLT (Section 5.5), except that transform does not specify a matrix A. It should therefore not be surprising that any Johnson-Lindenstrauss transform provides an oblivious subspace embedding, after some adjustment of parameters. The Count Sketch gives a weaker result than the Sparse JLT, but it can also be used for a subspace embedding via a different argument. The necessary size of the summary is somewhat larger, but we gain in terms of time cost: the Count Sketch is much faster to apply than even the Sparse JLT, due to its increased sparsity.

Given a subspace embedding, we can apply it immediately to approximately solve a regression problem. We apply the definition of the subspace embedding (equation 6.2) to the $n \times (d+1)$ matrix A' formed by concatenating A with b. This means for any vector y in the $d+1$-dimensional subspace spanned by A', we have $\|Sy\|_2^2 \in (1 \pm \epsilon)\|y\|_2^2$. In particular, we can consider only those y for which the final component is fixed to the value of -1. Rewriting, the problem of least squares regression is to find

$$\arg\min_x \|Ax - b\|_2^2 = \arg\min_{y:y_{d+1}=-1} \|A'y\|_2^2$$

By the above, we replace $A'y$ with $(SA')y$ and solve the (smaller) regression instance

$$\arg\min_{y:y_{d+1}=-1} \|(SA')y\|_2^2$$

using whatever method we prefer. For example, we could use the closed form $x = (A^T S^T S^T A)^{-1} A^T S^T b$, or apply gradient descent.

We can observe that this approach is quite general. For instance, the problem of constrained regression additionally imposes constraints on the solution. That is, $x \in C$ for some (usually convex) set C. Then the same approach works, since additionally constraining x does not affect the accuracy of S. Hence we just need to be able to solve the (smaller) constrained regression instance given by (SA) and (Sb) in place of A and b. For example, the popular LASSO regression model imposes the (L_1 regularization) condition $\|x\|_1 \leq \tau$, for some parameter τ. This can immediately be applied in this reduced dimensionality instance.

History and Background. The notion of using dimensionality reduction to speed up regression, and the demonstration of the subspace embedding property is due to Sarlós [198]. Clarkson and Woodruff showed that Count Sketch could give similar results in time proportional to the input sparsity [51]. Nelson and Nguyen, and Woodruff and Zhang variously give improved bounds for oblivious subspace embeddings and their applications [187, 230]. A more detailed discussion, including omitted technical proofs, is given by Woodruff [228].