CSE 392: Matrix and Tensor Algorithms for Data Spring 2024

Lecture 9 — February 12, 2024

Instructor: Shashanka Ubaru *Scribe: Anish Pandya*

Last class, we discussed the Gaussian embedding, with independent and identically distributed entires. We also discussed the subsampled Hadamard transform. This class, we focused on Countsketch, Sparse Sketching matrix, and sketch and solve for least squares regression, and went through some examples in MatLab.

Recall that a **Gaussian subspace embedding** works well for a small sketch size. Because $\mathbf{S} \in \mathbb{R}^{m \times n}$, we have to generate mn random, iid entries, and so computing the product $\mathbf{SA}$ takes $\mathcal{O}(mnd)$ time. We also discussed the **subsampled Hadamard transform** (SRHT), where $\mathbf{S} = \mathbf{PHD} \in \mathbb{R}^{m \times n}$. This algorithm is faster, since we can apply the produce $\mathbf{SA}$ in $\mathcal{O}(mn \log d)$ time. The disadvantage is that a larger sketch size is needed, and fast multiplication is only possible if $\mathbf{D}$ is dense enough. If $\mathbf{A}$ is sparse, it is harder because of the recursive structure of the Hadamard matrix.

Sparse Sketching Approach

In 2013 then, to address these problems, the **Sparse Embedding Approach** was proposed, adapted from CountSketch. In this, $\mathbf{S}$ has one ± 1 per column. Note that a row vector of the data matrix, $\mathbf{A}_i$ contributes $\pm \mathbf{A}_i$ to one of the columns of $\mathbf{SA}$.

Definition. Sparse sketching matrix. For $i \in [n]$, pick uniformly, and independently $h_i \in [m]$, $s_i \in \{-1, 1\}$, and define $\mathbf{S} \in \mathbb{R}^{m \times n}$ as:

$$\mathbf{S}_{h_i, i} \to s_i, i \in [n]$$
$$\mathbf{S}_{j,i} \to 0, \text{ else}$$

$\mathbf{s}$ is a sign vector, also known as a Rademacher vector. $\mathbf{h}$ hashes to the m buckets:

$$\mathbf{S}_{j*} = \sum_{i, h_i = j} s_i \mathbf{e_i}^T$$

Fact 1. Then it follows, to calculate the k-th column of the product: $\mathbf{SA}$, we need to calculate:

$$\begin{aligned} \mathbf{SA}_k &= \left(\sum_{i, h_i = j} s_i \mathbf{e}_i^T \right) \mathbf{A} \\ &= \sum_{i, h_i = j} s_i \mathbf{A}_i \end{aligned}$$

where, $\mathbf{A}_i$ are the indexed column vectors of $\mathbf{A}$.

The advantage is that to compute $\mathbf{SA}$, we only need to consider the number of non-zero entries of $\mathbf{A}$, that is the algorithm is $\mathcal{O}(nnz(\mathbf{A}))$time.

Fact 2. If $\mathbf{s}$ is a Rademacher (sign) vector, $\mathbf{s} \in \mathbb{R}^n$, and $\mathbf{x} \in \mathbb{R}^n$

$$\mathbb{E}[\mathbf{s}^T\mathbf{x}] = \|\mathbf{x}\|_2^2$$

Proof. If $\mathbf{s} \in \mathbb{R}^n$ is Rademacher, the entries s_i are drawn from the distribution $\rho(x_i) = \frac{1}{2}\delta(x_i - 1) + \frac{1}{2}\delta(x_i + 1)$, for $x_i \in \mathbb{R}$. Then from linearity of expectation, we see:

$$\begin{aligned}
\mathbb{E}\left[\mathbf{s}^T\mathbf{x}\right] &= \mathbb{E}\left[\sum_i^n s_i^T x_i\right] \\
&= \sum_i^n x_i \, \mathbb{E}\left[s_i^T\right] \\
&=
\end{aligned}$$

■

Analysis of Sparse Embeddings

Theorem 1. ***Variance of Countsketch.*** *For $\mathbf{S} \in \mathbb{R}^{m \times n}$, a sparse sketching distribution, and $\mathbf{y} \in \mathbb{R}^n$, a unit vector:*

$$Var\left(\|\mathbf{S}\mathbf{y}\|_2^2\right) \leq \frac{3}{m}$$

Proof. Let $\mathbf{z} = \mathbf{S}\mathbf{y}$. Then, from the norm preserving property of $\mathbf{S}$, we have: $\|\mathbf{z}\|_2^2 = \|\mathbf{S}\mathbf{y}\|_2^2 = \|\mathbf{y}\|_2^2 = 1$. Next, we need to compute: $\mathbb{E}\left[\|\mathbf{S}\mathbf{y}\|_2^4\right]$. By independence, many of the cross terms go to zero:

$$\begin{aligned}
\mathbb{E}\left[\|\mathbf{S}\mathbf{y}\|_2^4\right] &= \mathbb{E}\left[\left(\langle \mathbf{z}, \mathbf{z}\rangle\right)^2\right] \\
&= \mathbb{E}\left[\left(\sum_i z_j^2\right)^2\right] \\
&= \sum_{i,j} \mathbb{E}\left[z_i^2 z_j^2\right] \\
&= \sum_i \left(\mathbb{E}[z_i^4] - \mathbb{E}[z_i^2]^2\right) + \sum_{i \neq j} \mathbb{E}[z_i^2]\,\mathbb{E}[z_j^2] \\
&\leq 1 + \sum_i \mathbb{E}[z_i^4]
\end{aligned}$$

Then, we can substitute the definition of $\mathbf{z}$:

$$\sum_i \mathbb{E}[z_i^4] = \sum_{i,j,k,\ell} \mathbb{E}\left[s_i y_i s_j y_j s_k y_k s_\ell y_\ell\right]$$

The summand is only nonzero when two indicies are equal and the other two indicies are also equal (by indepenence and the first moment of the distribution). This only occurs three times (if $i = j$,

then $k = \ell$, if $i = k$, $j = \ell$, and if $i = \ell$, $k = j$). So then:

$$\sum_i \mathbb{E}[z_i^4] = 3 \cdot \frac{1}{m}\frac{1}{m}\|\mathbf{y}\|_2^4$$
$$= \frac{3}{m^2}$$

So then, from the previous sum, we see that this is just the result we wish to show:

$$\text{Var}\left(\|\mathbf{Sy}\|_2^2\right) \leq \frac{3}{m}$$

■

An important note made in lecture regarding this proof is the potential for overcounting and the necessity of careful accounting of the terms. Dr. Ubaru uses the idea of the hash function to note that if an index takes one value, then the other cannot happen, by the definition of a hash function.

In lecture, we also discussed that the important idea of the preceeding (and succeeding) proofs, and the course more generally, is that randomized algorithms have theoretical guarantees that can be applied to a diverse problem set. Dr. Ubaru also remarks that this is the goal of the course.

We then proceed to show that the Countketch matrix is a subspace embedding:

Theorem 2. *For $\mathbf{S} \in \mathbb{R}^{m\times n}$, a Countsketch matrix, and $\mathbf{A} \in \mathbb{R}^{n\times d}$, if $m = \mathcal{O}(d^2/(\delta\epsilon^2))$, then, with probability of at least $1-\delta$:*

$$\|\mathbf{SAx}\|_2 = (1+\epsilon)\|\mathbf{Ax}\|_2$$

Proof. As seen above $\mathbf{S}$ satisfies the $(\epsilon, \delta, 2)-$JL moment property (lecture 8), for a unit vector $\|\mathbf{y}\|_2 = 1$:

$$\mathbb{E}\left[|\|\mathbf{Sy}\|_2^2 - 1|^2\right] \leq \frac{3}{m} \leq \epsilon^2\delta$$

Then, we have following the previous result for $\ell = 2$, that: $m \geq 3/(\epsilon^2\delta)$, so $m = \mathcal{O}(1/(\epsilon^2\delta))$ satisfies the desired condition and so $\mathbf{S}$ is an ϵd subsapce embedding. ■

Now, let's reveiw some of the types of sketching matricies we have seen:

Sketching Matrix	Sketch Size m	Time-cost to compute $\mathbf{SA}$
JL, i.i.d Sub-Gaussians	$\mathcal{O}(d\log(1/\delta)/\epsilon^2)$	$\mathcal{O}(mnd)$
Fast JL -SRHT	$\mathcal{O}(d\log(1/\delta)\log(d)/\epsilon^2)$	$\mathcal{O}(mn\log d)$
Countsketch	$\mathcal{O}(d^2/(\delta\epsilon^2))$	$\mathcal{O}(\text{nnz}(\mathbf{A}))$

So, we can see that the Countsketch matrix is generally much worse in sketch size for most situations where $\mathbf{A}$ is not sparse. We remark that depending on your goal, the sketching matrix may differ. For example, if the goal is less storage and the computational (time) cost is less important, then the Gaussian sketch works well. If we have a dense matrix of data, then it makes sense to use SRHT to sketch $\mathbf{A}$. Lastly, in applications where we have sparse data and we want (quicker) computation, the asymptotic costs indicate to use Countsketch. Here we've outlined some of the advantages and disadvantages of the methods we've discussed in class and seen in the homework. It should be noted that we did not mention SNAPs.

Sketch and Solve Least Squares Regression.

Now, we turn to discussing an application of sketching to solving overdetermined least squares. Informally, we want to find the best $\mathbf{x}*$ such that $\mathbf{A}\mathbf{x}* \approx \mathbf{b}$. As we have seen, the exact solution is given by computing the Moore-Penrose Psuedoinverse. Either by QR factorization or the SVD, the cost for exactly computing the pseudoinverse if $\mathcal{O}(nd^2)$.

Can we come up with an approximate solution within $(1+\epsilon)$ that is way faster?

Let's consider the formal definition of the least squares regression problem, given in Dr. David Woodruf's monograph and in the lecture slides:

Definition. Given $\mathbf{A} \in \mathbb{R}^{n\times d}$, and $\mathbf{b} \in \mathbb{R}^n$, compute $\mathbf{x}^*$ such that:

$$\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2 \leq (1+\epsilon)\|\mathbf{A}\mathbf{x}^* - \mathbf{b}\|_2$$

for some error parameter $\epsilon \in \mathbb{R}$.

To use sketching, we would have to:

1. Generate a sketching matrix $\mathbf{S}$.
2. Compute the sketches: $\mathbf{SA}, \mathbf{Sb}$.
3. Solve:

$$\tilde{\mathbf{x}} = \min_{x\in\mathbb{R}^d} \|\mathbf{SA}\mathbf{x} - \mathbf{Sb}\|_2^2$$

Recall from the previous class the definition of a subspace embedding:

Definition. Embedding. A matrix $\mathbf{S}$ is an $\epsilon-$embedding of a set $\mathcal{P} \subset \mathbb{R}^n$ if, for all $\mathbf{y} \in \mathcal{P}$,

$$\|\mathbf{S}\mathbf{y}\|_2 = (1+\epsilon)\|\mathbf{y}\|_2$$

Definition. Subspace Embedding. Let $\mathbf{S} \in \mathbb{R}^{m\times n}$ have independent Gaussian entries. If $m = \mathcal{O}(d\log(1/\delta)/\epsilon^2)$, given $\mathbf{A} \in \mathbb{R}^{n\times d}$, with probability $1-\delta$, we have:

$$\|\mathbf{S}\mathbf{A}\mathbf{x}\|_2 = (1+\epsilon)\|\mathbf{A}\mathbf{x}\|_2$$

If we let $\mathbf{y} = \mathbf{A}\mathbf{x}$, we can now see, that this just means that for $\mathbf{S}$ to be a subspace embedding of $\mathbf{A}$, if $\mathbf{S}$ is an embedding for Range$\{\mathbf{A}\}$, for every $\mathbf{x} \in \mathbb{R}^d$.

Theorem 3. ***Sketch & Solve.*** *Suppose $\mathbf{S} \in \mathbb{R}^{m\times n}$ is a subspace embedding for span$([\mathbf{A}, b])$. Then, let:*

$$\mathbf{x}^* = \min_{\mathbf{x}\in\mathbb{R}^d} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2$$

Also, let $\tilde{\mathbf{x}}$ be the sketched solution:

$$\tilde{\mathbf{x}} = \min_{\mathbf{x}\in\mathbb{R}^d} \|\mathbf{S}(\mathbf{A}\mathbf{x} - \mathbf{b})\|_2$$

For $\epsilon \leq (1/3)$, we have:

$$\|\mathbf{A}\tilde{\mathbf{x}} - b\|_2 \leq (1+3\epsilon)\|\mathbf{A}\mathbf{x}^* - \mathbf{b}\|_2$$

Proof. Let $\mathbf{y} = [\mathbf{x}, -1]^T$, $\mathbf{x} \in \mathbb{R}^d$. Using the subspace embedding property, we have:

$$\begin{aligned}\|\mathbf{S}(\mathbf{Ax} - \mathbf{b})\|_2 &= \|\mathbf{S}[\mathbf{Ab}]\mathbf{y} \\ &= (1 \pm \epsilon)\|[\mathbf{Ab}]\mathbf{y}\|_2 \\ &= (1 \pm \epsilon)\|\mathbf{Ax} - b\|_2\end{aligned}$$

Then, from this, we can see:

$$\begin{aligned}\|\mathbf{A}\tilde{\mathbf{x}} - \mathbf{b}\|_2 &\leq \frac{\|\mathbf{SA}\tilde{\mathbf{x}} - \mathbf{Sb}\|_2}{1 - \epsilon} \\ &\leq \frac{\|\mathbf{SAx}^* - \mathbf{Sb}\|_2}{1 - \epsilon} \\ &\leq \frac{1 + \epsilon}{1 - \epsilon}\|\mathbf{Ax}^* - \mathbf{b}\|_2\end{aligned}$$

Therefore, we see from the binomial theorem and by the fact that $(1 - \epsilon) > (1 - 3\epsilon)$, $\epsilon \leq (1/3)$ that:

$$\|\mathbf{A}\tilde{\mathbf{x}} - b\|_2 \leq (1 + 3\epsilon)\|\mathbf{Ax}^* - \mathbf{b}\|_2$$

■

We then discussed the time complexity of the algorithm, which is on the order of the number of nonzero entries plus something on the order of (d^3/ϵ^2) or more. There was some discussion of improvemend via pre-conditioning and the conjugate gradient algorithm.