Randomized Linear Algebra

Krishna Balasubramanian

March 30, 2020

1 Introduction

Matrices (and tensors) play a crucial role in large-scale data analysis. A crucial aspect of dealing with such large-scale matrices is the use of randomization for performing even routine operations that otherwise become computationally prohibitive. For more details, refer to [Mah16].

1.1 Notation

For a matrix $\mathbf{A}$ a matrix of size $m \times n$,

- $\mathbf{A}_{i,:} \in \mathbb{R}^n$ denotes the i-th row
- $\mathbf{A}_{:,j} \in \mathbb{R}^m$ denotes the j-th column
- $A_{i,j}$ denote the entry at the (i,j)-th position.
- Matrix norms satisfy following properties:
  - $\|\mathbf{A}\| \geq 0$ and $\|\mathbf{A}\| = 0$ if and only if $A = 0$ (positivity)
  - $\|\alpha\mathbf{A}\| = |\alpha|\|\mathbf{A}\|$ (homogeneity)
  - $\|\mathbf{A} + \mathbf{B}\| \leq \|\mathbf{A}\| + \|\mathbf{B}\|$
- Frobenius norm: $\|A\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n \mathbf{A}_{i,j}^2}$
- Induced norm: Given a vector norm $\|\cdot\|$, we can define the corresponding **induced norm** by

$$\begin{aligned}\|\mathbf{A}\| &= \sup_x \{\|\mathbf{A}x\| : \|x\| = 1\} \\ &= \sup_x \left\{ \frac{\|\mathbf{A}x\|}{\|x\|} : x \neq 0 \right\}\end{aligned}$$

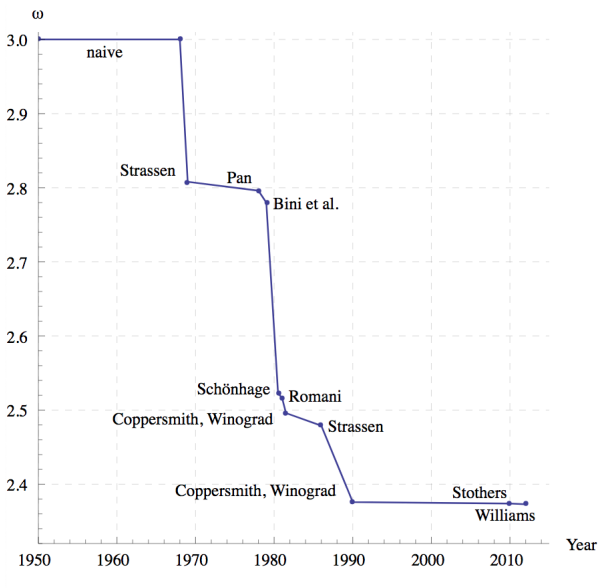

Figure 1: The bound on ω over time. Source; Wikipedia.

- Operator norm: $\|\mathbf{A}\|_2 = \sup_{x \neq 0} \frac{\|\mathbf{A}x\|_2}{\|x\|_2} = \sigma_{max}(\mathbf{A})$ (the maximum singular value). In this notes, we use $\|\mathbf{A}\|$ to denote $\|\mathbf{A}\|_2$, the operator norm.

Fact 1.0.1. Let $\mathbf{A} = ab^\top$ be a rank-1 matrix. Then $\|\mathbf{A}\|_F = \|\mathbf{A}\| = \|a\|_2\|b\|_2$. Prove it.

2 Randomized Matrix Multiplication

Given $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{n \times p}$ the problem we consider is to compute or approximate $\mathbf{C} = \mathbf{AB}$. A naive way of multiplying two matrices is given in Algorithm 1.

Algorithm 1 Vanilla Matrix Multiplication Algorithm

```
for i = 1, ..., m do
    for j = 1, ..., n do
        C_{i,j} = A_{i,:} B_{:,j} (= Σ_{k=1}^{n} A_{i,k} B_{k,j})
    end for
end for
Return C
```

Assuming $m = p = n$, the computational complexity of the above naive algorithm is $\mathcal{O}(n^\omega)$ with $\omega = 3$. Over the years, several researchers have worked to bring the exponent ω down 2.3728639. See Figure 1 for an overview. Such algorithms typically care about **exactly** computing the matrix $\mathbf{C}$. What if we are OK with **approximately** computing the matrix $\mathbf{C}$? It turns out, one can develop fast algorithms in this case. Its based on the following crucial observation: Algorithm 1 is based on inner-product operations, but another way to

$$\begin{aligned}
\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \begin{pmatrix} a & d \\ b & e \\ c & f \end{pmatrix} &= \begin{pmatrix} 1 \\ 4 \\ 7 \end{pmatrix} \begin{pmatrix} a & d \end{pmatrix} + \begin{pmatrix} 2 \\ 5 \\ 8 \end{pmatrix} \begin{pmatrix} b & e \end{pmatrix} + \begin{pmatrix} 3 \\ 6 \\ 9 \end{pmatrix} \begin{pmatrix} c & f \end{pmatrix} \\
&= \begin{pmatrix} 1a & 1d \\ 4a & 4d \\ 7a & 7d \end{pmatrix} + \begin{pmatrix} 2b & 2e \\ 5b & 5e \\ 8b & 8e \end{pmatrix} + \begin{pmatrix} 3c & 3f \\ 6c & 6f \\ 9c & 9f \end{pmatrix} \\
&= \begin{pmatrix} 1a+2b+3c & 1d+2e+3f \\ 4a+5b+6c & 4d+5e+6f \\ 7a+8b+9c & 7d+8e+9f \end{pmatrix}.
\end{aligned}$$

Figure 2: Outer Product view of matrix multiplication

do matrix multiplication directly is to view it as based on outer-product operations.

View $\mathbf{C} = \mathbf{AB}$ as sum of rank-one matrices (or outer products) as follows (see also figure 2):

$$\mathbf{C} = \mathbf{AB} = \sum_{i=1}^{n} \mathbf{A}_{:,i}\mathbf{B}_{i,:}$$

Note that each outer product in the above summation is a rank-1 matrix. Based on this, the idea is to *randomly* select r rank-one components, which leads to the randomized algorithm for matrix multiplication given in Algorithm 2.

Algorithm 2 Randomized Matrix Multiplication Algorithm

for $l = 1, \ldots, r$ **do**
  Pick $i_l \in \{1, \ldots, n\}$ i.i.d. with probability $\mathbb{P}\{i_l = k\} = p_k$
end for
Return

$$\mathbf{M} = \sum_{l=1}^{r} \frac{1}{rp_{i_l}} \mathbf{A}_{:,l}\mathbf{B}_{l,:}$$

First, note that $\mathbf{M}$ is *not a deterministic* matrix, it is a *random matrix*. From a statistical point of view, $\mathbf{M}$ is an **estimator** of the true matrix $\mathbf{C}$. Note that we have

$$\begin{aligned}
\mathbb{E}[\mathbf{M}] &= \sum_{l=1}^{r} \sum_{k} \mathbb{P}\{i_l = k\} \frac{1}{rp_k} \mathbf{A}_{:,k}\mathbf{B}_{k,:} \\
&= \sum_{k} \mathbf{A}_{:,k}\mathbf{B}_{k,:} \\
&= \mathbf{AB} = \mathbf{C}
\end{aligned}$$

So, $\mathbf{M}$ is an **unbiased** estimator of the true matrix $\mathbf{C}$. The error, for example $\|\mathbf{M} - \mathbf{C}\|_F$ depends on $\{p_k\}$. The probabilities $\{p_k\}$ are called as *importance sampling probabilities*.

So how to compute p_k ? Is there an *optimal* way of doing so ? Indeed, we can uniformly randomly select the outer products. In this case, the values are set to

$$p_k = \frac{1}{n}.$$

One can also do non-uniform sampling, where we set the values to

$$p_k = \frac{\|\mathbf{A}_{:,k}\|_2 \|\mathbf{B}_{k,:}\|_2}{\sum_l \|\mathbf{A}_{:,l}\|_2 \|\mathbf{B}_{l,:}\|_2} \tag{1}$$

In this case, p_k can be computed using one pass and $\mathcal{O}(n)$ memory. Intuitively this biases the sampled rank-1 components towards "large" rank-1 matrix. Here by "large" we mean the components that contribute most to towards the sum. There are two results we can provide about Algorithm 2 with the choice of importance sampling probabilities in Equation (1) which makes the use of this algorithm appealing in practice.

2.1 Choice of p_k

First, as it turns out the choice of importance sampling probabilities in Equation (1), minimizes $\mathbb{E}\left[\|\mathbf{M} - \mathbf{AB}\|_F^2\right]$. To see that, note since $\mathbb{E}[\mathbf{M}] = \mathbf{C}$, we have

$$\mathbb{E}\left[\|\mathbf{M} - \mathbf{AB}\|_F^2\right] = \mathbb{E}\left[\sum_{i,j} (\mathbf{M}_{i,j} - \mathbf{A}_{i,:}\mathbf{B}_{:,j})^2\right] = \sum_{i,j} \mathsf{Var}\left[\mathbf{M}_{i,j}\right] \tag{2}$$

$$= \frac{1}{r}\sum_k \sum_{i,j} \frac{\mathbf{A}_{i,k}^2 \mathbf{B}_{k,j}^2}{p_k} - \frac{1}{r}\sum_{i,j} \left(\mathbf{A}_{i,:}\mathbf{B}_{:,j}\right)^2 \tag{3}$$

$$= \frac{1}{r}\sum_k \frac{1}{p_k} \|\mathbf{A}_{:,k}\|_2^2 \|\mathbf{B}_{k,:}\|_2^2 - \frac{1}{r}\|\mathbf{AB}\|_F^2$$

In addition, by Cauchy-Schwarz inequality, we have for sequences α_k and p_k,

$$\left(\sum_k \sqrt{\alpha_k}\right)^2 \leq \left(\sum_k p_k\right)\left(\sum_k \frac{\alpha_k}{p_k}\right)$$

with equality attained if $p_k \propto \sqrt{\alpha_k}$. Note that, we have by definition, $\sum_k p_k = 1$. Hence, we have

$$\frac{1}{r}\sum_k \frac{1}{p_k} \|\mathbf{A}_{:,k}\|_2^2 \|\mathbf{B}_{k,:}\|_2^2 - \frac{1}{r}\|\mathbf{AB}\|_F^2 \geq \frac{1}{r}\left(\sum_k \|\mathbf{A}_{:,k}\|_2 \|\mathbf{B}_{k,:}\|_2\right)^2 - \frac{1}{r}\|\mathbf{AB}\|_F^2$$

where the lower bound is attained when $p_k \propto \|\mathbf{A}_{:,k}\|_2 \|\mathbf{B}_{k,:}\|_2$. Together with the constraint that $\sum_k p_k = 1$, we have the choice of p_k in Equation (1).

2.2 Justification for approximation

Next, we will show that the algorithm has less error, with **very high probability**.

Theorem 2.1. *Suppose*

$$p_k = \frac{\|\mathbf{A}_{:,k}\|_2 \|\mathbf{B}_{k,:}\|_2}{\sum_l \|\mathbf{A}_{:,l}\|_2 \|\mathbf{A}_{l,:}\|_2}.$$

Then, as long as

$$r = C \log n$$

we have

$$\frac{\|\mathbf{M} - \mathbf{AB}\|_F}{\|\mathbf{A}\|_F \|\mathbf{B}\|_F} \leq \sqrt{\frac{1}{C}} \tag{4}$$

with probability $1 - \mathcal{O}(n^{-10})$.

Remark 2.1.1. Lets say we require the relative error in the RHS of equation 4 to be 10^{-1} with high-probability. Then we just pick $C = 100$. For large matrices, that is n being order of millions, the above algorithm practically is very fast and give high-accuracy solution with probability ≈ 1.

Proof. **Before proving this theorem, you are required to understand the notation and the result in Theorem 2.2.**

The proof involves calculating the quantity V and R in Theorem 2.2 and applying the result in Equation 5.

Step 1: Calculating V and R:

Let

$$X_l = \sum_{k=1}^{n} \frac{1}{rp_k} \mathbf{A}_{:,k} \mathbf{B}_{k,:} \mathbb{1}\{i_l = k\}$$

Then we have $\mathbf{M} = \sum_{l=1}^{r} \mathbf{X}_l$. Using Fact 1.0.1, we also have

$$\begin{aligned} \|\mathbf{X}_l\| &\leq \max_k \left\{ \frac{1}{rp_k} \|\mathbf{A}_{:,k}\|_2 \|\mathbf{B}_{k,:}\|_2 \right\} \\ &= \frac{1}{r} \sum_{k=1}^{n} \|\mathbf{A}_{:,k}\|_2 \|\mathbf{B}_{k,:}\|_2 \\ &= R \end{aligned}$$

Next, again using Fact 1.0.1, we have

$$\begin{aligned}\mathbb{E}\left[\sum_{l=1}^{r}\|\mathbf{X}_l\|^2\right] &= r\sum_{k=1}^{n}\frac{\mathbb{P}\{i_l=k\}\|\mathbf{A}_{:,k}\|_2^2\|\mathbf{B}_{k,:}\|_2^2}{r^2p_k^2}\\ &\leq \frac{\left(\sum_{k=1}^{n}\|\mathbf{A}_{:,k}\|_2\|\mathbf{B}_{k,:}\|_2\right)^2}{r}\\ &= V\end{aligned}$$

Note that R and V differ only by a square factor in the numerator.

Step 2: Applying the result in Equation 5:

Based on the result in Step 1, by Equation 5, we have

$$\begin{aligned}\|\mathbf{M}-\mathbf{C}\|_F &\leq \sqrt{V\log n}+R\log n\\ &\leq \sqrt{\frac{\log n}{r}}\left(\sum_{k=1}^{n}\|\mathbf{A}_{k,:}\|_2\|\mathbf{B}_{k,:}\|_2\right)\\ &\leq \sqrt{\frac{\log n}{r}}\|\mathbf{A}\|_F\|\mathbf{B}\|_F\end{aligned}$$

where the last step follows by Cauchy-Schwarz. □

2.3 Matrix Concentration

You are not required to understand the following result in complete detail. Make sure you understand Equation (5) and how it is used in the proof of Theorem 2.1. If you are curious about the result, you can refer to [Tro15]. You are also encouraged to first read scalar version of Bernstein's inequality from wikipedia: scalar Bernstein's inequality.

Theorem 2.2 (Matrix Bernstein Inequality)**.** *Let* $\{\mathbf{X}_l\}_{l=1}^{L}\in\mathbb{R}^{d_1\times d_2}$ *be a sequence of independent zero-mean random matrices. Assume that each matrix satisfies* $\|X_l\|\leq R$*. Let*

$$V=\max\left\{\left\|\mathbb{E}\left[\sum_{l=1}^{L}\mathbf{X}_l\mathbf{X}_l^\top\right]\right\|,\left\|\mathbb{E}\left[\sum_{l=1}^{L}\mathbf{X}_l^\top\mathbf{X}_l\right]\right\|\right\}.$$

Then we have

$$\mathbb{P}\left\{\left\|\sum_{l=1}^{L}\mathbf{X}_l\right\|\geq\tau\right\}\leq(d_1+d_2)\exp\left(\frac{-2\tau^2/2}{V+R\tau/3}\right)$$

To understand the above concentration inequality, answer the following two questions:

1. What happens when τ is not too large ? Can you see we have $\exp\left(-\tau^2/V\right)$

2. What happens when τ is large ? Can you see we have $\exp(-\tau/R)$

Fact 2.2.1. From the above theorem, one can show that we also have, with probability $1 - \mathcal{O}\left((d_1 + d_2)^{-10}\right)$

$$\left\| \sum_{l=1}^{L} \mathbf{X}_l \right\| \leq \sqrt{V \log(d_1 + d_2)} + R \log(d_1 + d_2) \tag{5}$$

References

[Mah16] Michael W Mahoney, *Lecture notes on randomized linear algebra*, arXiv preprint arXiv:1608.04481 (2016).

[Tro15] Joel Tropp, *An introduction to matrix concentration inequalities*, Foundations and Trends® in Machine Learning **8** (2015), no. 1-2, 1–230.