CS 229r: Algorithms for Big Data Fall 2013

Lecture 14 — Thursday, Oct 17 2013

Prof. Jelani Nelson *Scribe: Aleksandar Makelov*

1 Overview

Today, we'll be looking at subspace embeddings, and how to use them to get fast algorithms for least squares regression. Next time we'll see how to use them for low-rank approximation as well.

2 Subspace embeddings

Recall from the end of last lecture that a *subspace embedding* is a linear transformation that has the Johnson-Lindenstrauss property for all vectors in the subspace:

Definition 1. *Given $W \subset \mathbb{R}^n$ a linear subspace and $\varepsilon \in (0,1)$, an ε-**subspace embedding** is a matrix $\Pi \in \mathbb{R}^{m \times n}$ for some m such that*

$$\forall x \in W : (1-\varepsilon)\|x\|_2 \leq \|\Pi x\|_2 \leq (1+\varepsilon)\|x\|_2$$

Claim 2. *For any linear subspace $W \subset \mathbb{R}^n$ with $\dim W = d$, there exists a 0-subspace embedding $\Pi \in \mathbb{R}^{d \times n}$, but no ε-subspace embedding $\Pi \in \mathbb{R}^{m \times n}$ with $\varepsilon < 1$ if $m < d$.*

Proof. For the $m = d$ case, first rotate the subspace W to become $\text{span}(e_1, \ldots, e_d)$ (via multiplication by an orthogonal matrix), and then project to the first d coordinates. This clearly preserves norms in W exactly.

Now, assume there is an ε-subspace embedding $\Pi \in \mathbb{R}^{m \times n}$ for $m < d$. Then, the map $\Pi : W \to \mathbb{R}^m$ has a nontrivial kernel, in particular there is some $w \in W, w \neq 0$ such that $\Pi w = 0$. On the other hand, $\|\Pi w\|_2 \geq (1-\varepsilon)\|w\|_2 > 0$, contradiction. □

How can we find the orthogonal matrix used in the proof efficiently? Suppose W is given to us in terms of a matrix $A \in \mathbb{R}^{n \times d}$ that spans its columns. Then recall from linear algebra that

Theorem 3 (Singular value decomposition)**.** *Every $A \in \mathbb{R}^{n \times d}$ has a "singular value decomposition"*

$$A = U \Sigma V^T$$

where $U \in \mathbb{R}^{n \times r}$ has orthonormal columns, $r = \text{rank}(A)$, $\Sigma \in \mathbb{R}^{r \times r}$ is diagonal with strictly positive entries on the diagonal, and $V \in \mathbb{R}^{d \times r}$ has orthonormal columns.

Imagine W is the column span of A with $r = d$. By completing U to a square orthogonal matrix, $U^{-1} = U^T$ is the rotation we need. As for efficiently finding the SVD, we have

Theorem 4 (Demmel, Dumitru, Holtz [3]). *In the setting of the previous theorem, we can approximate SVD well in time $\tilde{O}(nd^{\omega-1})$ where ω is the constant in the exponent of the complexity of matrix multiplication. Here the tilde hides logarithmic factors in n.*

We shall assume henceforth in this lecture that we can in fact find the SVD exactly in $\tilde{O}(nd^{\omega-1})$ time (to avoid talking about numerical analysis).

3 Least squares regression

3.1 Definition and motivation

Definition 5. *Suppose we're given $A \in \mathbb{R}^{n\times d}$, $b \in \mathbb{R}^n$ where $n \gg d$. We want to solve $Ax = b$; however, since the system is over-constrained, an exact solution does not exist in general. In the **least squares regression** problem, we instead want to solve the equation in a specific approximate sense: we want to compute*

$$x^* = \text{argmin}_x \|Ax - b\|$$

The choice of the function to be optimized is not arbitrary. For example, assume that we have some system, and one of its parameters is a linear function of d other parameters. How can we find the coefficients of that linear function? In reality, we experimentally observe a linear function + some random error. Under certain assumptions - errors have mean 0, same variance, and are independent, then least squares regression is provably the best estimator out of a certain class of estimators (see the Gauss-Markov theorem).

3.2 How do we solve least squares in general?

What's the best x to choose? Notice that $\{Ax : x \in \mathbb{R}^d\}$ is the column span of A. Then, the x^* we need is the projection of b on that column span. Let $A = U\Sigma V^T$ be the SVD of A. Then the projection of b satisfies

$$\text{Proj}_{\text{Col}(A)} b = UU^T b$$

hence we can set $x^* = V\Sigma^{-1}U^T b$ since then we have $Ax^* = U\Sigma V^T V\Sigma^{-1}U^T b = UU^T b$. Thus, we can solve LSR in $\text{SVD}(n, d)$ time.

3.3 Using subspace embeddings

We want to be faster than the above algorithm. Let Π be an ε-subspace embedding for the span of b and columns of A, which has dimension $\leq d + 1$. We'll compute $\tilde{x}^* = \text{argmin}_x \|\Pi Ax - \Pi b\|$, and want to show that $\tilde{x}^*$ is an approximate solution. Indeed,

$$(1-\varepsilon)\|A\tilde{x}^* - b\|_2 \leq \|\Pi A\tilde{x}^* - \Pi b\|_2 \leq \|\Pi Ax^* - \Pi b\|_2 \leq (1+\varepsilon)\|Ax^* - b\|_2$$

and thus $\|A\tilde{x}^* - b\|_2 \leq \frac{1+\varepsilon}{1-\varepsilon}\|Ax^* - b\|_2$, so we are within a $1 + O(\varepsilon)$ factor of the optimal solution. Thus, provided we have an efficient subspace embedding, we might be able to have a fast approximate algorithm for least squares regression.

3.4 The Fast JL Transform approach

How can we get such an efficient ε-subspace embedding? Recall that in Problem 1 of Problem set 5, we showed there is an ε-subspace embedding $\Pi \in \mathbb{R}^{m \times n}$ with $m = O(d/\varepsilon^2)$ that works with probability $1 - 1/e^{cd}$. This embedding may be very good, but getting there costs a lot: we have to multiply ΠA, which can be done in $O(mnd^{\omega-2})$ (when we break the matrices in $d \times d$ blocks) time, which is worse than $O(nd^{\omega-1})$ — the complexity of SVD on the original instance!

Sarlos [7] was the first to investigate how we can speed up the slow part of the algorithm discussed in the previous paragraph, and he used the fast Johnson-Lindenstrauss transform. The time to compute Πx in FJLT is $O(n \log n + m^3)$, and $m \approx d$, ignoring ε, so $\approx O(n \log n + d^3)$. So, ΠA can be found in time $O(nd \log n + d^4)$. We can use other results (Ailon, Liberty) on the FJLT to get rid of the additive d^4 for a slightly worse m.

3.5 Sparse embedding approach

Clarkson et al [2] showed we can get $m = \frac{d \log^6(d/\varepsilon)}{\varepsilon^2}$ with a matrix Π that has $s = 1$ non-zeroes per column.

Mahoney and Meng [4], and Nelson and Nguyễn [5] show that $m = O(d^2/\varepsilon^2)$ with $s = 1$ suffices. A possible approach to show this is the following: observe that Π being an ε-subspace embedding for a d-dimensional linear subspace $W \subset \mathbb{R}^n$ spanned by the columns of a matrix $A = U\Sigma V^T$ in singular value decomposition is equivalent to $\Pi U^T \Pi U - I_d$ having small operator norm:

$$\|\Pi U^T \Pi U - I_d\| \leq \varepsilon$$

up to changing ε by a factor of at most 2. To show this,we first do some linear algebra review.

3.5.1 Linear algebra review

Definition 6. *For a matrix A, the* **operator norm is** $\|A\| = \sup_{\|x\|=1} \|Ax\|$**.**

Claim 7. *If $A = U\Sigma V^T$ in SVD, the operator norm is the largest entry in Σ, and also equal to $\sqrt{\lambda_{max}(A^T A)}$. The vector realizing it is the corresponding column of V. The diagonal entries of Σ are called the* singular values *of A, and the least singular value $\sigma_{min}(A)$ satisfies*

$$\sigma_{min}(A) = \inf_{\|x\|=1} \|Ax\|.$$

This inf is also realized by the appropriate column of V. Also,

$$\|A\| = \sup_{\|x\|=1} |x^T Ax| \text{ if } A \text{ is symmetric}$$

3.5.2 Proving the equivalence.

Applying the above facts to our case for the symmetric $\Pi U^T \Pi U - I_d$, if the operator norm is

$$\begin{aligned}\varepsilon > \|\Pi U^T \Pi U - I_d\| &= \sup_{\|x\|=1} |x^T (\Pi U)^T (\Pi U) x - x^T x| \\ &= \sup_{\|x\|=1} \|\|\Pi U x\|^2 - 1|\end{aligned}$$

then the lengths of all unit norm vectors in W are preserved up to error ε, and thus the lengths of all vectors in W are preserved up to error ε.

This might remind you of approximate matrix multiplication (AMM). Recall from last lecture that approximate matrix multiplication has a guarantee of the form

$$\mathbb{P}\left(\|\Pi U^T \Pi U - U^T U\|_F > \varepsilon \|U\|_F^2\right) < \delta.$$

Since the operator norm is at most the Frobenius norm, this implies

$$\mathbb{P}\left(\|\Pi U^T \Pi U - U^T U\| > \varepsilon \|U\|_F^2\right) < \delta.$$

By orthonormality of the columns of U, we have $\|U\|_F^2 = d$. Hence, we can apply AMM with error ε/d. Then the sparse Thorup-Zhang sketch [8] from Problem 3, Problem set 1 uses $m = O(d^2/\varepsilon^2)$, so we're done. The fact that AMM already implies that the Thorup-Zhang sketch gives sparse subspace embeddings was observed by Huy L. Nguyễn. Π has one nonzero per column, so ΠA can be multiplied very quickly (in time linear in the number of non-zeroes of A); on the other hand, the SVD takes more time since we're doing it for d^2/ε^2 (there are some tricks around this by applying the FJLT after the Thorup-Zhang sketch; see [2]). It's also possible to get $m = d^{1.01}/\varepsilon^2$ by increasing s to $s = O(1/\varepsilon)$ [5].

4 Other ways to use subspace embeddings

4.1 Iterative algorithms

This idea is due to Tygert and Rokhlin [6] and Avron et al. [1]. The idea is to use gradient descent. The performance of the latter depends on the *condition number* of the matrix:

Definition 8. *For a matrix A, the **condition number** of A is the ratio of its largest and smallest singular values.*

Let Π be a $1/4$ subspace embedding for the column span of A. Then let $\Pi A = U \Sigma V^T$ (SVD of ΠA). Let $R = V \Sigma^{-1}$. Then by orthonormality of U

$$\forall x : \|x\| = \|\Pi A R x\| = (1 \pm 1/4)\|A R x\|$$

which means $AR = \tilde{A}$ has a good condition number. Then our algorithm is the following

1. Pick $x^{(0)}$ such that

   $$\|\tilde{A} x^{(0)} - b\| \leq 1.1 \|\tilde{A} x^* - b\|$$

   (which we can get using the previously stated reduction to subspace embeddings with ε being constant).

2. Iteratively let $x^{(i+1)} = x^{(i)} + \tilde{A}^T(b - \tilde{A}x^{(i)})$ until some $x^{(n)}$ is obtained.

We will give an analysis following that in [2] (though analysis of gradient descent can be found in many standard textbooks). Observe that

$$\tilde{A}(x^{(i+1)} - x^*) = \tilde{A}(x^{(i)} + \tilde{A}^T(b - \tilde{A}x^{(i)}) - x^*) = (\tilde{A} - \tilde{A}\tilde{A}^T\tilde{A})(x^{(i)} - x^*),$$

where the last equality follows by expanding the RHS. Indeed, all terms vanish except for $\tilde{A}\tilde{A}^T b$ vs $\tilde{A}\tilde{A}^T\tilde{A}x^*$, which are equal because x^* is the optimal vector, which means that x^* is the projection of b onto the column span of $\tilde{A}$.

Now let $AR = U'\Sigma'V'^T$ in SVD, then

$$\begin{aligned}
\|\tilde{A}(x^{(i+1)-x^*})\| &= \|(\tilde{A} - \tilde{A}\tilde{A}^T\tilde{A})(x^{(i)} - x^*)\| \\
&= \|U'(\Sigma' - \Sigma'^3)V'^T(x^{(i)} - x^*)\| \\
&= \|(I - \Sigma'^2)U'\Sigma'V'^T(x^{(i)} - x^*)\| \\
&\leq \|I - \Sigma'^2\| \cdot \|U'\Sigma'V'^T(x^{(i)} - x^*)\| \\
&= \|I - \Sigma'^2\| \cdot \|\tilde{A}(x^{(i)} - x^*\| \\
&\leq \frac{1}{2} \cdot \|\tilde{A}(x^{(i)} - x^*)\|
\end{aligned}$$

by the fact that $\tilde{A}$ has a good condition number. So, $O(\log 1/\varepsilon)$ iterations suffice to bring down the error to ε. In every iteration, we have to multiply by AR; multiplying by A can be done in time proportional to the number of nonzero entries of A, $\|A\|_0$, and multiplication by R in time proportional to d^2. So the dominant term in the time complexity is $\|A\|_0 \log(1/\varepsilon)$, plus the time to find the SVD.

4.2 Yet another approach

This approach is due to Sarlós [7]. First, a bunch of notation: let

$$\begin{aligned}
x^* &= \text{argmin}\|Ax - b\| \\
\tilde{x}^* &= \text{argmin}\|\Pi Ax - \Pi b\|. \\
A &= U\Sigma V^T \text{ in SVD} \\
Ax^* &= U\alpha \text{ for } \alpha \in \mathbb{R}^d \\
Ax^* - b &= -w \\
A\tilde{x}^* - Ax^* &= U\beta
\end{aligned}$$

Then, $OPT = \|w\| = \|Ax^* - b\|$. We have

$$\begin{aligned}
\|A\tilde{x}^* - b\|^2 &= \|A\tilde{x}^* - Ax^* + Ax^* - b\|^2 \\
&= \|A\tilde{x}^* - Ax^*\|^2 + \|Ax^* - b\|^2 \text{ (they are orthogonal)} \\
&= \|A\tilde{x}^* - Ax^*\|^2 + OPT^2 = OPT^2 + \|\beta\|^2
\end{aligned}$$

We want $\|\beta\|^2 \leq 2\varepsilon OPT^2$. Since $\Pi A, \Pi U$ have same column span,

$$\begin{aligned}\Pi U(\alpha+\beta) = \Pi A\tilde{x}^* &= \text{Proj}_{\Pi A}(\Pi b) = \text{Proj}_{\Pi U}(\Pi b) \\ &= \text{Proj}_{\Pi U}(\Pi(U\alpha + w)) = \Pi U\alpha + \text{Proj}_{\Pi U}(\Pi w)\end{aligned}$$

so $\Pi U\beta = \text{Proj}_{\Pi U}(\Pi w)$, so $(\Pi U)^T(\Pi U)\beta = (\Pi U)^T\Pi w$. Now, let Π be a $(1 - 1/\sqrt[4]{2})$-subspace embedding – then ΠU has smallest singular value at least $1/\sqrt[4]{2}$. Therefore

$$\|\beta\|^2/2 \leq \|(\Pi U)^T(\Pi U)\beta\|^2 = \|(\Pi U)^T\Pi w\|^2$$

Now suppose Π also approximately preserves matrix multiplication. Notice that w is orthogonal to the columns of A, so $U^T w = 0$. Then, by the general approximate matrix multiplication property,

$$\mathop{\mathbb{P}}_{\Pi}\left(\|(\Pi U)^T\Pi w - U^T w\|_2^2 > \varepsilon'^2\|U\|_F^2\|w\|_2^2\right) < \delta$$

We have $\|U\|_F = \sqrt{d}$, so set error parameter $\varepsilon' = \sqrt{\varepsilon/d}$ to get

$$\mathbb{P}\left(\|(\Pi U)^T\Pi w\|^2 > \varepsilon\|w\|^2\right) < \delta$$

so $\|\beta\|^2 \leq 2\varepsilon\|w\|^2 = 2\varepsilon OPT^2$, as we wanted.

So in conclusion, we don't need Π to be an ε-subspace embedding. Rather, it suffices to simply be a c-subspace embedding for some fixed constant $c = 1 - 1/\sqrt{2}$, while also providing approximate matrix multiplication with error $\sqrt{\varepsilon/d}$. Thus for example using the Thorup-Zhang sketch, using this reduction we only need $m = O(d^2 + d/\varepsilon)$ and still $s = 1$, as opposed to the first reduction in these lecture notes which needed $m = \Omega(d^2/\varepsilon^2)$.

References

[1] Haim Avron and Petar Maymounkov and Sivan Toledo. Blendenpik: Supercharging LAPACK's least-squares solver *SIAM Journal on Scientific Computing*, 32(3) 1217–1236, 2010.

[2] Kenneth L. Clarkson and David P. Woodruff. Low rank approximation and regression in input sparsity time *Proceedings of the 45th Annual ACM Symposium on the Theory of Computing (STOC)*, 81–90, 2013.

[3] James Demmel, Ioana Dumitriu, and Olga Holtz. Fast linear algebra is stable. *Numer. Math.*, 108(1):59-91, 2007.

[4] Xiangrui Meng and Michael W. Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression *Proceedings of the 45th Annual ACM Symposium on the Theory of Computing (STOC)*, 91–100, 2013.

[5] Jelani Nelson and Huy L. Nguyễn. OSNAP: Faster numerical linear algebra algorithms via sparser subspace embeddings. *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2013.

[6] Vladimir Rokhlin and Mark Tygert. A fast randomized algorithm for overdetermined linear least-squares regression. *Proceedings of the National Academy of Sciences*, 105 (36) 13212–13217, 2008.

[7] Tamas Sarlós. Improved approximation algorithms for large matrices via random projections. *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 143–152, 2006.

[8] Mikkel Thorup, Yin Zhang. Tabulation-Based 5-Independent Hashing with Applications to Linear Probing and Second Moment Estimation. *SIAM J. Comput.* 41(2): 293–331, 2012.