CS369M: Algorithms for Modern Massive Data Set Analysis **Lecture 5 - 10/07/2009**

Approximating L2 regression

Lecturer: Michael Mahoney *Scribes: Kshipra Bhawalkar and Mengqiu Wang*

** Unedited notes.*

1 Introduction

All of the methods we have seen so far have provided an additive error guarantee i.e. the approximate matrix A' is such that,

$$|A - A_k| \leq |A - A'| \leq |A - A_k| + \epsilon|A|$$

In this lecture we study some methods that provide relative guarantees i.e. guarantees of the form,

$$|A - A'| \leq (1 + \epsilon)|A - A_k|$$

Relative error guarantees are a relatively new thing, for many years it was not even known if such guarantees could be achieved. But once people were able to generate such guarantees many more emerged some notable guarantees include DMM, HP using geometric algorithms, DV using iterative update, Sarlos for fast matrix computations. One notable property of these algorithms is that they are not efficiently implementable in few passes.

In this lecture we study a relative error guarantee for approximately solving the problem of linear regression. More formally given a matrix $A \in \mathbb{R}^{n \times d}$ and a vector $b \in \mathbb{R}^d$, we want to find a vector $x \in \mathbb{R}^n$ such that $Ax = b$. It's important to note that such an x may not always exist, hence we find the best possible solution i.e. find x s.t.

$$x = arg\ min_x \|Ax - b\|_2 \ and\ z = min_x \|Ax - b\|_2$$

2 Solving linear regression in $O(nd^2)$ time

The linear regression can be solved exactly in $O(nd^2)$ time using various decompositions of A like QR and SVD. We now look at some randomized method which calculate the correct answer in $O(nd^2)$ time with high probability. Note that these methods are clearly worse than solving the system exactly but we present them here as they will be used as the basis for algorithms in the next section.

Approach 1: Random Projection
We generate a random matrix $R \in \mathbb{R}^{l \times n}$, as in previous lectures and then solve,

$$min_x \|R^T Ax - R^T b\|_2$$

This gives a $(1 + \epsilon)$ approximation and takes $O(nd^2)$ time.

Approach 2: Random Sampling
We could carry out random sampling similar to in previous lectures. Unfortunately, this does not work very well. Matrix A might loose rank in the random sampling and it is hard to capture that in a relative error term. To illustrate this better consider a matrix A with $n - 1$ rows of vector α and one row containing a vector α' which is much different from α. Then any sort of sampling would loose α' with high probability and we will have high error. Remember that in the previous lecture we had additive error for random sampling which is why loosing rank was not an issue.

Q: How can we still make random sampling work?
Idea: De-convolute subspace of A and size of A information.

Let $A = U\Sigma V^T$ be the singular value decomposition of A. We use these decomposition to define modified sampling probabilities. Define $\{P_i\}_{i=1}^n : P_i = \frac{\|(U_A)_{(i)}\|_2^2}{\sum_i' \|(U_A)_{(i')}\|_2^2} = \frac{\|(U_A)_{(i)}\|_2^2}{d}$.

Intuition for $\|(U_A)_{(i)}\|_2^2$:
the values $\|(U_A)_{(i)}\|_2^2$ correspond to the diagonal elements of the projection matrix P_A.

$$(P_A)_{ij} = (A(A^TA)^TA^T)_{ij} = (UU^T)_{ij} = \langle (U_A)_{(i)}, (U_A)_{(j)} \rangle$$

These values allow us to detect points in the data set that might have disruptive effect or might have high leverage on the model. There are also referred as leverage scores or effective resistance. Real data often have a high variety in leverage scores, so these values allow us to detect those early on.

Now we propose the modified random sampling algorithm:
Algorithm: Sampling L_2 algorithm.

- Compute $P_i = \frac{1}{d}\|(U_A)_{(i)}\|_2^2$.
- Sample $O(\frac{d \lg d}{\epsilon^2})$ rows and generate sampling matrix S.
- Solve $min_x|SAx - Sb|$

This algorithm still takes $O(nd^2)$ just for the first step. It is an open question as to whether the P_i's can be computed approximately with low relative error in $o(nd^2)$ time.

In the next section we analyze the error rate of this algorithm and discuss how it can be modified for faster performance.

3 Analysis of the modified Random Sampling Algorithm

Theorem 3.1 *If $\{P_i\}$ s.t. $P_i \geq \beta.\frac{\|(U_A)_{(i)}\|_2^2}{d}$, $\beta \in (0,1)$, $r \geq \theta(k^2/\beta\epsilon)$, then with high probability*

$$\|b - A\tilde{x}_{OPT}\| \leq (1+\epsilon)z = (1+\epsilon)\|b - Ax_{OPT}\| \tag{1}$$

$$\|x_{OPT} - \tilde{x}_{OPT}\| \leq \epsilon\kappa(A)\sqrt{\gamma^2 - 1}\|x\|_2 \tag{2}$$

where, $\gamma \in (0,1)$ is such that, $\|U_AU_A^Tb\| \geq \gamma\|b\|$.

Remark 3.2 *The k^2 in the form for r can be improved to $k \lg k$.*

Before proceeding to prove the theorem stated above we establish the following structural result.

Lemma 3.3 *Let $\mathcal{X}$ be any matrix such that*

$$\sigma_min(\mathcal{X}U_A) \geq 9/10 \tag{3}$$

$$\|U_A^T\mathcal{X}^T\mathcal{X}b^\perp\| \leq \epsilon/2z^2 \tag{4}$$

$$then (*1) and (*2) holds \tag{5}$$

where $b^\perp$ is the part of b outside the space spanned by A i.e. $b^\perp = b - A^+Ab$

Proof: $\min_{X'} \|\mathcal{X}_b - \mathcal{X}A_{X'}\| = \min_{y'} \|\mathcal{X}Ax_{OPT} + b^{\perp} - \mathcal{X}Ax_{OPT} + y'\|$ hence:

$$\begin{aligned} b &= Ax_{OPT} + b^{\perp} && (6)\\ &= \min_{y'} \|\mathcal{X}b^{\perp} - \mathcal{X}Ay'\| && (7)\\ &= min_{z'}\|\mathcal{X}b^{\perp} - \mathcal{X}U_A Z'\| ** && (8)\\ & && (9)\end{aligned}$$

where $Z' = \Sigma U^T y'$

Define: $Z = \Sigma_A V_A^T(x_{OPT} - \tilde{x}_{OPT})$ note: this is an optimal solution to ** Solve ** using normal equation $(\mathcal{X}U_A)^T(\mathcal{X}U_A)Z = (\mathcal{X}U_A)^T x b^{\perp}$ take the norm on both sides

$1/2|Z|^2 \leq |(\mathcal{X}U_A)^T(\mathcal{X}U_A)Z| \leq Z^2/2$

look at residual vector

$$\begin{aligned} & |b - Ax_{OPT}^2|_2^2 = |b - Ax_{OPT}^2 + Ax_{OPT} - Ax_{OPT}|_2^2 && (10)\\ = \quad & |b - A^2 x_{OPT}|_2^2 + |A(x_{OPT} - Ax_{OPT}^2|_2^2 && (11)\\ = \quad & Z^2 + |U_A Z|_2^2 && (12)\\ \leq \quad & Z^2 + \epsilon Z^2 && (13)\\ = \quad & (1+\epsilon)Z^2 && (14)\\ & && (15)\end{aligned}$$

to get second claim: $A(x_{OPT} - \tilde{x}_{OPT}) = U_A Z$ where $Z = \Sigma_A V_A^T(x_{OPT} - \tilde{x}_{OPT})$

$$\begin{aligned} \|x_{OPT} - \tilde{x}_{OPT}\|_2^2 &\leq \frac{|U_A Z|_2^2}{\sigma_{min}^2(A)} && (16)\\ &\leq \frac{\epsilon Z^2}{\sigma_{min}^2(A)} *** && (17)\\ & && (18)\end{aligned}$$

Assume $|U_A U_A^T b|_2 \geq \gamma |b|$ Then:

$$\begin{aligned} Z^2 &= |b|_2^2 - |U_A U_A^T b|_2^2 && (19)\\ &\leq (\gamma^{-2} - 1)|U_A U_A^T b|_2^2 && (20)\\ &\leq \sigma_{max}(A)(\gamma^{-2} - 1)|x_{OPT}| && (21)\\ & && (22)\end{aligned}$$

Since:

$$\begin{aligned} x_{OPT} &= A^{-1}b && (23)\\ &= V\Sigma^{-1}U^T b && (24)\\ |x_{OPT}| &= |V\Sigma^{-1}U^T b| && (25)\\ &\geq \sigma_{min}(\Sigma^{-1})|U^T b| && (26)\\ &= \frac{|U_A U_A^T b|^2}{\sigma_{max}(A)} && (27)\\ & && (28)\end{aligned}$$

Play it back to ***, we get:

$$\begin{aligned} \|x_{OPT} - \tilde{x}_{OPT}\|_2^2 &\leq \frac{\epsilon Z^2}{\sigma_{min}^2(A)} && (29)\\ &\leq \epsilon K(A)^2 \sqrt{\gamma^{-2}}|x_{OPT}| && (30)\\ & && (31)\end{aligned}$$

Note: I may have lost a square root on ϵ

Key Assumption: $\mathcal{X}U_A$ is something like an orthogonal matrix $XU_A \perp Xb^{\perp}$

Remark 3.4 *If S is construct such that $p_i \geq \beta \cdot |(U_A)_{(i)}|_2^2/d$ then S satisfies the conditions on $\mathcal{X}$ in the above lemma.*

Extension 1: Relative error low rank matrix approximation.
Given a matrix A. Let $p_i := |U_{A,k}|_2^2/k$. Sample $r = O(d^2)$ columns of A and return C, then

$$\|A - P_c A\|_2 \leq (1+\epsilon)\|A - A_k\|_2$$

Extension 2: Fast Johnson Lindenstrauss Transform (Ailon, Chazelle)
Algorithm for fast least squares.

- Preprocess A,b by randomized Hadamard transform
- Uniformaly sample $O(d \lg d/\epsilon)$ rows.
- Solve induced subproblem.

then $(1+\epsilon)$ approximation in $o(nd^2)$ time.

definition: **Hadamard Matrix**
A hadamard matrix is a matrix with entries from $\{-1, 1\}$, whose rows and columns are orthonormal. A family of hadamard matrices for $n = 2^k$ can be constructed recursively as follows,

$$H_n = \begin{pmatrix} H_{n/2} & H_{n/2} \\ H_{n/2} & -H_{n/2} \end{pmatrix} \text{ and} H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Any hadamard matrix can be normalized by multiplying with $1/\sqrt{n}$.

Note
For a hadamard matrix H, Hx for any $x \in \mathbb{R}^n$ can be computed in $O(nlog(n))$ time.

We hope to use hadamard matrices to generate smoother versions of vector x. Towards that we define a measure $\|x\|_\infty/\|x\|_2$ as the measure of localization and set out to construct H with low localization. With deterministic H, can have dense vectors x s.t. Hx is sparse. Hence we used a randomized hadamard transform HD, where D is a diagonal matrix with entries from $\{-1, 1\}$. The D needs to be chosen randomly so that HDx will have low localization with high probability.

Theorem 3.5 *(Ailon, Chazelle 2006) [1] If x is a unit vector, $\mathcal{H} = HD$ where H is a normalized hadamard matrix and D is a random diagonal matrix with entries from $\{-1, 1\}$. Let $y = \frac{1}{\sqrt{n}}\mathcal{H}x$ then $\|y\| = 1$ and with high probability $\|y\|_\infty = \theta(\frac{\log n}{n})$.*

Theorem 3.6 *[2] Let U be an $n \times d$ orthogonal matrix, let $N = max\{n, d\}$ and let $\mathcal{H} = HD$ be the $n \times n$ randomized Hadamard transform described above. Then with high probability,*

$$|(\mathcal{H}U)_{ij} \leq C_0\sqrt{\frac{logN}{n}}$$

$$|(\mathcal{H}U)_{(i)}|_2^2 \leq C_0 \frac{dlogN}{n}$$

for some constant C_0.

Now we define the modified algorithm that will run in $o(nd^2)$ time.
The algorithm calculates a solution x'_{opt} for the problem,

$$\mathcal{Z} = min_{x \in \mathbb{R}^d} |\mathcal{H}(Ax - b)|_2$$

Since $\mathcal{H}$ is orthonormal this will also be a solution to the original problem. The algorithm is as follows,

Algorithm

- Define a sampling matrix S and the hadamard transform.
- Preprocess the data with hadamard transform $\mathcal{H}$ i.e. calculate $\mathcal{H}A$, $\mathcal{H}b$
- Sample r rows uniformly where $r = O(dlogd/\epsilon)$.
- Solve $min|\mathcal{S}\mathcal{H}U\Sigma V^T - \mathcal{S}\mathcal{H}b|$.

More details and proofs can be found in [2]. See bibliography for other related papers.

References

[1] N. Ailon and B. Chazelle, Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform, STOC, pages 557-563, 2006

[2] P. Drineas, M. W. Mahoney, S. Muthukrishnan, T. Sarlos, Faster Least Squares Approximation.

[3] P. Drineas, M. W. Mahoney, S. Muthukrishnan, 'Sampling Algorithms for ℓ_2 Regression and applications.

[4] Rokhlin, Tygert, 'A fast randomized algorithm for overdetermined linear list-squares regression'.

[5] Avron, Maymounkov, and Toledo, "Blendenpik: Supercharging LAPACK's least-squares solver'.