Stat260/CS294: Randomized Algorithms for Matrices and Data

Lecture 21 - 11/18/2013

Lecture 21: Low-rank Approximation with Element-wise Sampling, Cont.

Lecturer: Michael Mahoney *Scribe: Michael Mahoney*

Warning: these notes are still very rough. They provide more details on what we discussed in class, but there may still be some errors, incomplete/imprecise statements, etc. in them.

21 Low-rank Matrix Approximation with Element-wise Sampling, Cont.

We continue with the discussion from last time. There is no new reading, just the same as last class.

In particular, today we will cover the following topics.

- A more sophisticated version of the element-wise sampling algorithm that is roughly comparable to the additive-error column/row sampling algorithms.
- Introductory discussion of the extension of element-wise sampling algorithm to the matrix completion problem.

21.1 A More Sophisticated Version of the Element-wise Sampling Algorithm

Let's start by describing a more sophisticated version of the element-wise random sampling algorithm we discussed last time. This is interesting in and of itself as well as since it will enable us to make a connection with the additive-error random sampling algorithm.

Again, we'll follow the motivation from AM07, which describes it nicely. Observe that in the algorithm from last class that when we sparsify the matrix A by keeping every entry with the same probability p, then we get

$$\mathbf{Var}\left[\hat{A}_{ij}\right] = \frac{1-p}{p} A_{ij} \quad \forall i, j.$$

In particular, this means that smaller entries of A lead to random variables with smaller variance. On the other hand, the bound on $\left\|A - \hat{A}\right\|_2$ depends on the maximum variance. Thus, to improve the results, one idea is to keep entries of A_{ij} with probability $p_{ij} \leq p$, so that all entries in $\hat{A}$ have roughly the same variance. This will help us to get sparser matrices, while keeping similar quality-of-approximation bounds.

In more detail, if we choose

$$\hat{A}_{ij} = \begin{cases} A_{ij}/p_{ij} & \text{with probability } p_{ij} \\ 0 & \text{otherwise} \end{cases},$$

then

$$\begin{aligned}\mathbf{E}\left[\hat{A}_{ij}\right] &= A_{ij} \\ \mathbf{Var}\left[\hat{A}_{ij}\right] &= A_{ij}^2\left(\frac{1}{p_{ij}}-1\right).\end{aligned}$$

In this case, if $p_{ij} = p\frac{A_{ij}^2}{b^2}$. where $p \in (0,1]$ and $b = \max_{ij}|A_{ij}|$, then

$$\mathbf{Var}\left[\hat{A}_{ij}\right] = \frac{b^2}{p} - A_{ij} \approx \frac{b^2}{p},$$

$\forall i,j$. With this choice of p_{ij}, we have that

$$\mathbf{E}\left[\text{ number of entries kept }\right] = \sum_{ij} p_{ij} = \sum_{ij} p\frac{A_{ij}^2}{b^2} = \frac{p}{b^2}\|A\|_F^2 = \frac{b}{p^2}mn\text{Avg}(A_{ij})^2 \ll pmn,$$

where pmn is what is obtained for uniform sampling.

There is the technical issue here that complicates things, but it is a real issue, and that is the so-called range constraint, i.e., the bound on the range that the random variable is allowed to take. The issue if that if we choose a very low-probability element, then we must rescale it by a lot, and this might violate the range constraint. (Many in the improvements since AM07 have to do with fixing this issue; we will summarize these below.) If we let

$$p_{ij} = \max\left\{\tau_{ij}, \sqrt{\tau_{ij}\frac{8\left(\log(n)\right)^4}{n}}\right\},$$

where $\tau = p\left(A_{ij}/b\right)^2$, then we have the following theorem.

Theorem 1 *Let $A \in \mathbb{R}^{m\times n}$, with $76 \le m \le n$, and let $b = \max_{ij}|A_{ij}|$. For $p > 0$, let*

$$\begin{aligned}\tau_{ij} &= \frac{p}{b^2}A_{ij}^2 \\ p_{ij} &= \max_{ij}\left\{\tau_{ij}, \tau_{ij}\frac{8\left(\log(n)\right)^4}{n}\right\}.\end{aligned}$$

Let $\hat{A} \in \mathbb{R}^{m\times n}$ be a random matrix with entries identically distributed as

$$\hat{A}_{ij} = \begin{cases} A_{ij}/p_{ij} & \textit{with probability } p_{ij} \\ 0 & \textit{otherwise}\end{cases}.$$

Then, we have the following.

- *W.p. $\ge 1-\exp\left(-19\left(\log(n)\right)^4\right)$, the matrix $\Delta = A - \hat{A}$ satisfy the following:*

$$\begin{aligned}\|N_k\|_2 &< 4b\sqrt{n/p} \\ \|N_k\|_F &< 4b\sqrt{kn/p}.\end{aligned}$$

- $\mathbf{E}\left[nnz(\hat{A}\right] \leq \frac{p}{b^2}\left\|A\right\|_F^2 + m\left(8\log(n)\right)^4 = pmn \cdot Avg\left[\left(\frac{A_{ij}}{b}\right)^2\right] + m\left(8\log(n)\right)^4$, *where* $Avg\left[\left(\frac{A_{ij}}{b}\right)^2\right] = \frac{\|A\|_F^2}{mn}$.

Proof: Choosing $p_{ij} = \max\{\cdot,\cdot\}$ ensures that the range constraint is not violated, since no probability is too small. Since $p_{ij} \leq \tau_{ij} + \frac{(8\log(n))^4}{n}$, using p_{ij} instead of τ_{ij} adds no more than $mn\frac{(8\log(n))^4}{n}$ elements of $\hat{A}$ in expectation. The rest of the proof is similar to what we did before.

$\diamond$

Remark. We have bounded the number of nonzero entries in the sampled matrix in terms of various parameters, but we haven't actually showed that it is small, e.g., in the sense of guaranteeing that it is not larger than a pre-specified value. We will get to his soon.

Given the result in Theorem 1, let's ask the following questions.

- How long does it take to compute this approximation?
- How does this approximation compare with the previous additive-error and relative-error bounds that we had with column/row sampling?
- Can these results be improved/extended, either in worse-case or under assumptions on the input?

Running time. To answer the question about running time, we will now give an algorithm that, given n and any $s > 0$, in one pass over the data produces a matrix $\hat{A}$ with probability $p = \frac{sb^2}{\|A\|_F^2}$, s.t. $\mathbf{E}\left[\text{nnz}\left(\hat{A}\right)\right] \leq sm\left(8\log(n)\right)^4$. That is, it is efficient in the Pass Efficient Model, since it uses only 2 passes over the data and roughly linear in $m+n$ additional space and time.

Note that, given n, b, and fixed p, it is easy to do non-uniform sampling in a single pass over the data using probabilities

$$p_{ij} = \max\left\{\tau_{ij}, \tau_{ij}\frac{(8\log(n))^4}{n}\right\},$$

with $\tau_{ij} = p\left(A_{ij}/b\right)^2$. This gives $\mathbf{E}\left[\text{nnz}\left(\hat{A}\right)\right] \leq \frac{p\|A\|_F^2}{b^2} + m\left(8\log(n)\right)^4$. This is problematic if we want to implement the algorithm since the first term is not known at the outset, and that might correspond to more entries than we want to keep. To remedy this will involve a slight re-parameterization. In particular, to make this term $= s$, for a pre-specified s, we can let $p = \frac{sb^2}{\|A\|_F^2}$. To compute this takes one full pass over the data (which is acceptable in the pass efficient model). Here is the entire algorithm to do this.

This SAMPLE(s,n) algorithm takes as input s and n, and it does the following.

1. Let Q be an empty priority queue, and let $z = 0$
2. For all A_{ij}, do the following.
   (a) $z \leftarrow z + A_{ij}^2$.
   (b) Choose a number r_{ij} u.a.r. from $[0,1]$.

```
    (c) Insert $A_{ij}$ into $Q$ with key $\kappa_{ij} = \max\left\{ \frac{sA_{ij}^2}{r_{ij}}, \frac{sA_{ij}^2}{r_{ij}^2}\frac{(8\log(n))^4}{n} \right\}$
    (d) Remove from $Q$ all the elements with key smaller than $z$
3. Return the contents of $Q$.
```

Given this Sample(s, n) algorithm, here is a lemma regarding its performance.

Lemma 1 *Let $A \in \mathbb{R}^{m\times n}$, with $76 \leq m \leq n$. Then,* Sample*(s, n) yields a matrix $\hat{A}$ such that*

- *w.p. $\geq 1 - \exp\left(-19\left(\log(n)\right)^4\right)$, we have that the matrix $\Delta = A - \hat{A}$ is s.t.*

$$\begin{aligned} \|\Delta_k\|_2 &\leq 4\sqrt{\frac{n}{s}}\|A\|_F \\ \|\Delta_k\|_F &\leq 4\sqrt{\frac{kn}{s}}\|A\|_F \end{aligned}$$

- $\mathbf{E}\left[nnz\left(\hat{A}\right)\right] \leq s + m\left(8\log(n)\right)^4$

Proof: Let $p = \frac{sb^2}{\|A\|_F^2}$, and define $\tau_{ij} = p\left(\frac{A_{ij}}{b}\right)^2 = s\left(\frac{A_{ij}}{\|A\|_F}\right)^2$. Then, by Theorem 1, it suffices to show that each entry of A_{ij} is kept by Sample(s, n) w.p. $= p_{ij} = \max\left\{\tau_{ij}, \tau_{ij}\frac{(8\log(n))^4}{n}\right\}$. But an element A_{ij} is in Q when Sample(s, n) terminates iff

$$\frac{sA_{ij}^2}{r_{ij}} \geq \|A\|_F \quad \text{or} \quad \frac{sA_{ij}^2}{r_{ij}^2}\frac{\left(8\log(n)\right)^4}{n} \geq \|A\|_F .$$

But this is equivalent to $r_{ij} \leq p_{ij}$ and each r_{ij} chosen uniformly over $[0, 1]$.

$\diamond$

Comparison with column/row sampling algorithms. To answer the question about comparing this element-wise sampling algorithm with the previous column/row sampling methods, observe the following. Given a matrix A, we can call Sample(s, n), where $s = \frac{16n}{\epsilon^2}$ to get a matrix $\hat{A} \in \mathbb{R}^{m\times n}$, where $\hat{A}$ has $O\left(n/\epsilon^2 + m\left(\log(n)\right)^4\right)$ non-zeros, sampled from the distribution of the main theorem, with $p = \frac{16nb^2}{\epsilon^2\|A\|_F^2}$. This $\tilde{O}\left(m+n\right)$ dependency corresponds roughly to keeping a small number of columns/rows. Then, by the structural lemma and the main theorem, w.p. $\geq 1 - \exp\left(-19\left(\log(n)\right)^4\right)$, we have that

$$\begin{aligned} \left\|A - \hat{A}_k\right\|_2 &\leq \|A - A_k\|_2 + \epsilon\|A\|_F \\ \left\|A - \hat{A}_k\right\|_F &\leq \|A - A_k\|_F + 3\sqrt{\epsilon}k^{1/4}\|A\|_F \end{aligned}$$

So, the element-wise sampling algorithm, when the sampling is done with probability distribution that is proportional to the entries-squared, gives additive-error bounds, where the scale of the additive error is $\|\cdot\|_F$ (just as with the additive-error version of the column/row sampling algorihtms). (Thus, the results are very similar to the additive-error column/row sampling algorithms, but they are not directly comparable, due to some minor differences; see AM07 for the details.)

Improving/extending these results. To answer the question about possible extensions and improvements, observe that there are several types of extensions of this basic set up that are interesting.

- **Dealing with small entries.** We followed AM07, where smaller entries are sampled with higher probability, but there were complexities having to do with the range constraint, etc. Similar but slightly stronger results can be gotten by dealing with smaller entries in other ways. For example: Drineas-Zouzias-11 zero out sufficiently small entries; Achlioptas-Karnin-Liberty-13 sample with respect to $|A_{ij}|$ and get bounds with respect to the stable rank; and Kundu-Namirajan-Drineas-13 sample with respect to a convex combination of a distribution that is proportional to the elements and a distribution that is proportional to the square of the elements. We are not going to go though these results, although they lead to improvements in some of what we have discussed, and there are likely additional improvements that can be made. They also lead to somewhat simpler results, due to recent developments in matrix concentration bounds.

- **Getting better bounds.** We should note something about the structure of the bounds and what one might hope to achieve. We might hope for bounds on $\left\|A - \hat{A}\right\|_2$ and $\left\|A - \hat{A}_k\right\|_2$; and although we have bounds for $\left\|A - \hat{A}_k\right\|_F$, we can't expect to get interesting bounds for $\left\|A - \hat{A}\right\|_F$ (basically since we have zeroed out most of the elements of the input matrix, which leads to a large Frobenius norm difference between the two matrices). This is true, even though the low-rank approximations to them are close with respect to the Frobenius norm, which is all that is needed to apply the sparsified matrix in the context of an iterative algorithm, which was the original motivation.

  One might wonder what happens if we make additional assumptions on the matrix, e.g., what if the matrix is *exactly* rank k? In that case, i.e., when the matrix is exactly rank k, then relative-error approximation algorithms like we have for column/row sampling mean that the error is ϵ times 0, meaning that the matrix is reconstructed exactly. That is, if the matrix is exactly rank k, then one can use the leverage-based sampling algorithms or a random projection algorithm to reconstruct the matrix exactly. (Of course, in that case, the column/row-wise sampling is less interesting, since in that case *any* set of exactly k columns can be used to reconstruct the matrix—numerical and robustness issues aside.)

- **Matrix Completion.** This leads to the question of whether we can get relative error approximations and/or reconstruct the matrix exactly using element-wise sampling. The short answer is No, or Probably No, in general, but the answer is Yes under some assumptions (that are rather strong, but that are difficult to remove). This work has gone on subsequent to but somewhat-independently of the AM07 developments in TCS, and it goes by the name "matrix completion." This it is loosely motivated by recommendation systems problems, and several key papers in the area are due to Candes-Recht-09; Candes-Tao-10; Recht-11; and Gross-11.

BTW, the question of exactly versus approximately low-rank for element-wise methods is actually quite non-trivial. Reconstructing or well-approximating a matrix with element-wise sampling is a much harder problem than with column/row sampling/projection methods, and getting relative error or stronger additive error guarantees is challenging. In general, where there are such results, one needs to make rather strong assumptions to obtain them. Here are two examples of assumptions that people have used to get better results with element-wise sampling.

- **For Matrix Completion.** Here, the goal is to impute missing entries from an unobserved matrix. The assumption on the input is that that the coherence properties are nice, e.g., flat leverage scores, and that data are presented ii.d.; or that the coherence properties are arbitrary but that the data are presented in a way that conform to that; etc.
- **For Laplacian Solvers.** Here, the goal is to sample entires from an adjacency matrix to sparsity the graph to get faster solvers. The assumption on the input is that the matrix is not arbitrary but has additional structure such as that it is the adjacency matrix or the Laplacian matrix of a graph. In this case, one can take advantage of the graph theoretic structure to get interesting results. The reason is basically that the entries of the Laplacian matrix correspond to rows/columns of the edge-incidence matrix, and thus careful element-wise sampling of the matrix corresponds to column/row sampling of a different matrix.

For the next two classes we will deal with the first of these topics, and for the two classes after that we will deal with the second of these two topics.

21.2 Introduction to matrix completion

Since we have a bit of time left, we will cover some of the introduction to matrix completion now. During the next two classes, we will get into more detail.

The so-called matrix completion problem has to do with imputing or predicting the entries of a matrix from a partially observed version of the matrix. Since the problem is ill-posed, one needs to make assumptions on the matrix. This need not have anything to do with sampling elements, e.g., one could do other randomized or deterministic things, but—motivated by recommendation systems applications—one popular variant of the problem does consider element-wise sampling.

We will be covering the results in the following two papers.

- Recht, "A Simpler Approach to Matrix Completion"
- Chen, Bhojanapalli, Sanghavi, and Ward, "Coherent Matrix Completion"

As I mentioned above, the basic connection between these papers and the AM07 TCS-style results is that the element-wise sampling algorithm we discussed gives additive-error algorithms for arbitrary input (just as did row/column sampling when the sampling was with respect row norms and not leverage scores). In general, to get relative-error algorithms is much harder, e.g., one needs the matrices to be exactly low rank, one needs very strong incoherence assumptions, one needs to use tools from convex optimization, etc. But, if one has a relative-error approximation, then under the assumption that the matrix is exactly low-rank, the matrix is reconstructed exactly.

(That is, a relative-error algorithms gives the *exact* answer if the matrix is *exactly* low-rank. The reason is that the additional error is ϵ times a scale which equals 0. That's not the way most people think about it, but it's true, and it's helpful to know that to understand the connections between RandNLA algorithms and recent work on matrix completion. Of course, in the particular context of reconstructing exactly rank k matrices, that is not so helpful, since one just needs to choose *any* set of k linearly independent columns. And the AM07 algorithm is not helpful, since it is an additive-error algorithm.)

So far, we have considered a problem parameterized roughly as follows. Given a matrix A, sample entires to construct a matrix $\hat{A}$ such that $\tilde{A} \approx A$ in some sense (i.e., we have adopted mostly the algorithmic perspective). To understand the connections with recent work on matrix completion, we need to ask a different question (more like the statistical perspective) with the following parameterization. Given an unseen matrix A that satisfies some niceness assumptions, and assuming that I see entries from in some way, reconstruct A, exactly or approximately.

The simplest thing to do is to make strong assumptions on A and the element presentation. For example, assume that A is "nice" in that the leverage scores, cross leverage scores, etc., are uniform, and assume that the elements of A are presented u.a.r. and also assume that A is exactly rank k, then we can call a black box (that involves convex optimization, in the form or ℓ_1 minimization or some related nuclear norm minimization) and we solve the problem. There are of course extensions of this basic setup to exactly low-rank plus a few point-wise spikes, to very small additional noise, etc., but that is the basic setup.

We should note that the incoherence assumption is an extremely strong assumption and that the assumption that the entries are sampled i.i.d. is also extremely strong (e.g., with respect to plausible data generative processes in the motivating application domains). The CBSW paper describes a variant of this approach to matrix completion that is designed to relate to some of what we have been discussing.

- A will still need to be exactly rank k
- A can be arbitrary, i.e., there is no niceness assumption
- The elements of A are presented according to a complicated leverage-based distribution.

Some comments on this approach.

- The point is that this coherence-based or leverage-based approach of CBSW is a strictly stronger result, since the assumption of data presentation essentially takes care of the problem with bad cases with respect to niceness. In particular, we can assume that A is nice with respect to its coherence properties, in which case the complicated leverage-based distribution becomes the uniform distribution.
- This approach is more like the structural approach we have been following, where it highlights the key structural property. Then, if one wants to make strong assumptions about data presentation and matrix niceness one can, but this approach makes explicit the relationship between the two. Moreover, this approach makes it easier to relate this more statistical perspective to the more algorithmic perspective we have been following.