CS 6220: Data-Sparse Matrix Computations
Lecture 4

Lecturer: Anil Damle
Scribe: Aneesh Heintz

Lecture Date: Feb 4, 2020

1 Fixed-rank approximation problem continued...

Continuing from the last lecture, we are given $A \in \mathbb{R}^{m\times m}$, a target rank, k, and oversampling integer, q. The algorithm that solves the fixed-rank approximation problem is as follows:

1. Draw $\Omega \in \mathbb{R}^{n\times(k+p)}$ with iid $\mathcal{N}(0,1)$ entries.
2. Form the matrix product $Y = A\Omega$
3. Construct a matrix $Q \in \mathbb{R}^{m\times(k+p)}$ whose columns form an orthonormal basis for the range of Y through QR decomposition: $Y = QR$, where $R \in \mathbb{R}^{(k+p)\times n}$

However, now we wish to improve the upper bounds accuracy of randomized algorithms. Here, we apply the randomized sampling scheme to the matrix $B = (AA^T)^q A$, where $q > 0$ is a small integer. B has the same singular vectors as A, but its singular values decay more quickly. If we form a matrix product $Z = B\Omega$, then

$$\mathbb{E}||(I - P_Z)||_2 \leq [1 + \frac{\sqrt{k}}{p-1} + \frac{e\sqrt{k+p}}{p}\sqrt{n}]^{\frac{1}{2q+1}}\sigma_{k+1}$$

where $\mathbb{E}$ is the expectation w.r.t the random test matrix and σ_{k+1} is the (k + 1)th singular value of A. This amendment to the previous algorithm is quite similar to the following one. Here, we have $A = A^T, A \in \mathbb{R}^{n\times n}$, where

$$A = \begin{pmatrix} V_1 & V_2 \end{pmatrix} \begin{pmatrix} \Lambda_1 & \\ & \Lambda_2 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix}$$

We can use randomized subspace iteration to compute an orthonormal matrix $Q \in \mathbb{R}^{n\times k}$ whose range approximates the range of A:

for *k = 1, 2, ...* **do**
  Z = A $Q^{(k-1)}$
  Z = $Q^{(k)}R \rightarrow$ QR factorization
  T = $Q^{(k)T}AQ^{(k)} \rightarrow$ (Use $Q^{(k)}$ to project A into smaller space.)
  T = U $\Lambda U^T \rightarrow$ eigen decomposition
end

Under mild assumptions, $Q^{(k)}U \to$ (converges to) eigvecs of A, U_1 and $\Lambda \to \Lambda_1$. The computational bottleneck of randomized subspace iteration, however, is that computing $Y = A\Omega \to$ costs $T_{mult}(A)(k+p)$. Therefore, we can use a structured random matrix that allows us to pick Ω s.t. $A\Omega$ is "fast." One example of such way of choosing Ω is with the subsampled random Fourier transform (SRFT). Here, we pick $\Omega = DFS$. D is a signed Identity with random signs. F is the discrete Fourier transform matrix. $S \in \mathbb{R}^{n\times(k+p)}$ is a random subsampling without replacement from the columns of the $n \times n$ identity matrix.

Up until now, we assumed that we could choose Q s.t. Q^TA was meaningful. We knew before $A - QQ^TA$ was small. Think of Q^TA as a dimension reduction of the cols of A. The columns of Q^TA are low-dimensional representations of cols of A. Q^TAe_i is a low-dimensional (k + p) embedding of Ae_i that depends on A. Can we build or use the embedding of the columns of A that don't know anything about A?

Definition: For a matrix $G \in \mathbb{R}^{n\times k}$, it is an ϵ-accurate subspace embedding for A if

$$(1-\epsilon)||Ax||_2 \leq ||G^TAx||_2 \leq (1+\epsilon)||Ax||_2 \forall x$$

Given one of the subspace embedding, what is it good for? What can we do with it? One example is that we can solve the least squares problem

2 Least Squares

Given a matrix $A \in \mathbb{R}^{m\times n}$ and a vector $b \in \mathbb{R}^m$, the least squares problem solves the problem

$$\min_{x\in\mathbb{R}^n} ||Ax-b||_2^2$$

We can use randomized algorithms to find high-precision solutions to the linear least squares problem that are over- or under-determined and possibly rank deficient. Say G is good for $\begin{pmatrix} A & b \end{pmatrix}$. Therefore we can find $\hat{x}$ s.t. $||G^T(A\hat{x}-b)||_2$ is small. This would mean $||A\hat{x}-b||_2$ is also small, but we lose a factor of ϵ. In the grand scheme of things, it does not result in too much of a loss in accuracy. We could also "estimate" A^TB as A^TGG^TB (embed into lower dimension) because we preserve distances, inner products, etc. However, let's first tackle the least squares problem. Pick G to be independent of A to help solve least squares. Now, given $A \in \mathbb{R}^{m\times n}$ with $m >> n$ and $b \in \mathbb{R}^m$, want to solve

$$\min_x \frac{1}{2}||Ax-b||_2^2$$

If we assume A has full column rank, there are two algorithms to solve the least squares problem:

1. Blendenpik by Avron, Maymounkor, and Toledo in 2010
2. LSRN by Meng, Saunders, and Mahony in 2014

If we want to solve the least squares problem, one way is to use krylov subspace methods, such as the method of conjugate gradients (CG) or Minres, to solve $Mx = b$ for symmetric, positive-definite matrix M. Mathematically, we can solve $\min_x \frac{1}{2}||Ax - b||_2^2$ by solving $(\star)A^TAx = A^Tb$. We can solve $(\star)$ by forming applying CG or Minres to the normal equation.

LSQR is equivalent to applying CG on normal equations.
LSMR is equivalent to applying Minres on normal equations.

Instead of directly forming a Krylov subspace associated with A^TA, we can use Golub-Kahan bidiagonalization. For lecture purposes, we wanted to note the existence of LSQR & LSMR. Generally, it is hard to predict the number of iterations for CG-like methods. Therefore, when using LSQR to solve the least squares problem, we have that

$$\frac{||x^{(k)} - x^\star||_{A^TA}}{||x^{0)} - x^\star||_{A^TA}} \leq 2(\frac{\sqrt{k(A^TA)} - 1}{\sqrt{k(A^TA)} + 1})^k$$

$x^{(k)}$ is the guess for $x^\star$ at iteration k. $x^\star$ is the true solution. $x^{(0}$ is the initial guess. Absent any info about A, this produces an upper bound of the guess.

For least squares problems, there are two types of faster preconditions. To solve $\min_x ||Ax - b||_2$:

1. Left solve $M^TBx = M^Td$ for some M
2. Right solve $BNy = d$ for some $N \to x = Ny$

Theorem: $x^\star_{right} = Ny^\star$ where $y^\star$ solves (min length solution)

$$\min_y ||ANy - b||_2$$

If range(N) = range(A^T), then $x^\star = x^\star_{right}$, where $x^\star$ is the solution of $\min_x ||Ax - b||_2$.

3 References

1. Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM review, 53(2):217–288, 2011.

2. Saunders, Michael Mahoney, Michael. (2011). LSRN: A Parallel Iterative Solver for Strongly Over- or Underdetermined Systems. SIAM Journal on Scientific Computing. 36. 10.1137/120866580.