NYU CS-GY 9223D: Algorithmic Machine Learning and Data Science Fall 2020

Homework 3

Name: R. Teal Witter

Problem 1

Collaborators: None.

1. Using the gradient descent algorithm we analyzed in class and rearranging as described in the problem set up, we can get a vector $y^{(i)}$ such that

$$y^{(i)} - y^* = (I - \frac{1}{\lambda_1} A^T A)(y^{(0)} - y^*)$$

in $O(ndi)$ time. In order to get the desired $x^{(q)}$, we sum with the proper coefficients for the polynomial we want:

$$\begin{aligned} x^{(q)} - x^* &= c_0(y^{(0)} - y^*) + c_1(y^{(1)} - y^*) + \cdots + c_q(y^{(q)} - y^*) \\ &= \left[c_0(I) + c_1(I - \frac{1}{\lambda_1} A^T A) + \cdots + c_q(I - \frac{1}{\lambda_1} A^T A)^q \right] (y^{(0)} - y^*) \\ &= p(I - \frac{1}{\lambda_1} A^T A)(x^{(0)} - x^*). \end{aligned}$$

Notice that the initial starting vector $x^{(0)} = y^{(0)}$ and that the problem doesn't change so the optimal solution $x^* = y^*$. Thus we can substitute $(x^{(0)} - x^*)$ for $(y^{(0)} - y^*)$ between the second and third lines. If we save each $y^{(i)}$ as we go and use it for the $i+1$th iteration, we can calculate our $x^{(q)}$ in total time $O(ndq)$.

2. Choose $\gamma = \lambda_d / \lambda_1$. By Claim 4 in the Lanczos notes, there is a degree $q = O(\sqrt{1/\gamma} \log(1/\epsilon))$ polynomial p such that $p(1) = 1$ and $|p(t)| \leq \epsilon$ for $0 \leq t \leq 1 - \gamma$.

   We check that such a p is the one we want. Indeed, $p(1) = 1$ tells us the coefficents sum to 1 which is exactly what we want.

   The eigenvalues of $A^T A$ are $\lambda_1, \ldots, \lambda_d$ so the eigenvalues of $1/\lambda_1 A^T A$ are $\lambda_1/\lambda_1, \ldots, \lambda_d/\lambda_1$ and the eigenvalues of $M = I - 1/\lambda_1 A^T A$ are $1 - \lambda_1/\lambda_1, \ldots, 1 - \lambda_d/\lambda_1$.

   Then the eigenvalues of $p(M)$ are $p(1-\lambda_1/\lambda_1), \ldots, p(1-\lambda_d/\lambda_1)$. To see this, observe that the ith term of M for $0 \leq i \leq q$ is $c_i M^i = c_i V \Lambda^i V^T$ where $M = V \Lambda V^T$. So the jth eigenvalue of M is $p(1 - \lambda_j/\lambda_1)$.

   The smallest eigenvalue is $p(1-\lambda_1/\lambda_1) = p(0)$. The top eigenvalue of M is $p(1-\lambda_d/\lambda_1)$ since $\lambda_d \leq \lambda_j$ for all other eigenvalues λ_j of $A^T A$. Sure enough $p(1 - \lambda_d/\lambda_1) = p(1 - \gamma) \leq \epsilon$. This p is the one we want!

3. In $q = O(\sqrt{\lambda_1/\lambda_d} \log(1/\epsilon))$ iterations of matrix multiplication, we can get a matrix $p(I - 1/\lambda_1 A^T A)$ with top eigenvalue bounded above by ϵ.

Problem 2

Collaborators: Kelly Marshall.

1. Define $f(x) = \tilde{\lambda} x^T x - x^T A^T A x$. We want to show that $f(x)$ is a convex function. Let $x, y \in \mathbb{R}^d$ and $s \in [0, 1]$. Then

$$\begin{aligned}(1-s)f(x) + sf(y) &= (1-s)\tilde{\lambda} x^T x - (1-s)x^T A^T A x + s\tilde{\lambda} y^T y - s y^T A^T A y \\ &\geq (1-s)^2 \tilde{\lambda} x^T x - (1-s)^2 x^T A^T A x + s^2 \tilde{\lambda} y^T y - s^2 y^T A^T A y \qquad (1)\end{aligned}$$

since $0 \leq s \leq 1$ and $f(x) \geq 0$ imply that $(1-s)f(x) \geq (1-s)^2 f(x)$ and $sf(y) \geq s^2 f(y)$. Notice that $f(x) \geq 0$ follows from the supposition that v_1 is the arg min of $f(x)$ since $v_1 A^T A v_1 = \lambda_1 \leq \tilde{\lambda}$ by assumption. Continuing from Equation 1,

$$= \tilde{\lambda}(1-s)x^T(1-s)x - (1-s)x^T A^T A(1-s)x + \tilde{\lambda} s y^T s y - s y^T A^T A s y \quad = f((1-s)x + sy)$$

by linearity. Therefore $f(x)$ is convex.

To see that S is not convex, choose vectors e_1 and e_2 and scalar $1/2$. Both $||e_1||_2^2 = ||e_2||_2^2 = 1$ so $e_1, e_2 \in S$ but $||1/2 e_1 + 1/2 e_2||_2^2 = 1/4 + 1/4 = 1/2 < 1$ so the linear combination $1/2 e_1 + 1/2 e_2 \notin S$.

2. To show that projected gradient descent with $f(x)$ and S is exactly equivalent to the power method, we need to show that the update rules are the same and the final vector in the power method is the arg min over all the vectors.

We can write

$$f(x) = \tilde{\lambda} \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} \left(\sum_{j=1}^{d} a_{i,j} x_j \right)^2$$

where $a_{i,}$ is the ith row of A. Then

$$\begin{aligned}\frac{\partial f}{\partial x_j} &= 2\tilde{\lambda} x_j - \sum_{i=1} 2 a_{i,j} (a_{i,} \cdot x) = 2\tilde{\lambda} x_j - 2 a_{,j}^T (Ax) \\ \nabla f(x) &= 2\tilde{\lambda} x - 2 A^T A x. \qquad (2)\end{aligned}$$

For $x(i) - \eta \nabla f(x^{(i)}) = A^T (A z^{(i)})$, we choose $\eta = 1/(2\tilde{\lambda})$.

To see that the projection step is the same as normalizing, simply recall that S is the set of vectors with norm greater than or equal to 1.

The arg min returned by projected gradient descent is the same as the final vector $z^{(T)}$ from the analysis in class that each z moves closer and closer to v_1.

3. Define $g(x) = -(x^T A^T A x)/(x^T x)$. Then

$$\begin{aligned}\nabla g(x) &= -\frac{x^T x 2 A^T A x - x^T A^T A x 2x}{(x^T x)^2} \\ &= \frac{2}{(x^T x)^2} \left(x^T A^T A x x - x^T x A^T A x \right).\end{aligned}$$

For any scaling c and right singular vector v_i, $cv_i^T A^T Acv_i = c^2\sigma_i^2$ and $A^T Av_i = c\sigma_i^2 v_i$. Therefore $c^3 v_i^T A^T Av_i v_i = c^2\sigma_i^2 cv_i = c^3 v_i^T vA^T Av_i$ so $\nabla g(cv_i) = 0$ and cv_i must be a stationary point.

To see that g is non-convex, choose vectors v_d and v_1. Then

$$g(v_d) - g(v_1) = -\sigma_d^2 + \sigma_1^2 > 0 = \nabla g(v_d)^T (v_d - v_1)$$

so g is not convex.

4. Choose $z = v_1$. Then

$$g(v_i + tv_1) = -\frac{(v_i + tv_1)^T A^T A(v_i + tv_1)}{||v_i + tv_1||_2^2} = \frac{-\sigma_i^2 - t\sigma_1^2}{1+t}$$

because of the orthonormal property of v_i and v_1. Since $\sigma_1 > \cdots > \sigma_d$ by assumption,

$$g(v_i + tv_1) = \frac{-\sigma_i^2 - t\sigma_1^2}{1+t} < \frac{-\sigma_i^2 - t\sigma_i^2}{1+t} = g(v_i).$$

Problem 3

Collaborators: Indu Ramesh.

1. We sum the rows and columns of D. First observe that

$$\begin{aligned} D_{i,j} &= ||x_i - x_j||_2^2 \\ &= (x_i - x_j)^T (x_i - x_j) \\ &= ||x_i||_2^2 + ||x_j||_2^2 - 2x_j^T x_i. \end{aligned}$$

Then

$$\begin{aligned} \sum_{j=1}^{n} \sum_{i=1}^{n} D_{i,j} &= \sum_{j=1}^{n} \sum_{i=1}^{n} ||x_i||_2^2 + ||x_j||_2^2 - 2x_j^T x_i \\ &= \sum_{j=1}^{n} \sum_{i=1}^{n} ||x_i||_2^2 + \sum_{i=1}^{n} \sum_{j=1}^{n} ||x_j||_2^2 + \sum_{j=1}^{n} -2x_j^T \sum_{i=1}^{n} x_i \\ &= 2n \sum_{i=1}^{n} ||x_i||_2^2 \end{aligned}$$

where we use the assumption that $\sum_{i=1}^{n} x_i$ is the all 0s vector. Therefore $\sum_{i=1}^{n} ||x_i||_2^2 = ||D||_F^2/(2n)$. We simply iterate and sum over D in time $O(n^2)$ where D is $n \times n$.

2. We begin by summing each row

$$\begin{aligned} ||D_{i,}||_2^2 &= \sum_{j=1}^{n} ||x_i||_2^2 + ||x_j||_2^2 - 2x_i^T x_j \\ &= n||x_i||_2^2 + \sum_{j=1}^{n} ||x_j||_2^2 - 0. \end{aligned}$$

Then the difference of the sums of two rows yields

$$||D_{i,}||_2^2 - ||D_{k,}||_2^2 = n(||x_i||_2^2 - ||x_k||_2^2).$$

Fix i. We sum

$$\begin{aligned} \sum_{k=1}^{n} D_{i,k} + \frac{||D_{i,}||_2^2 - ||D_{k,}||_2^2}{n} &= \sum_{k=1}^{n} 2||x_i||_2^2 - 2x_i^T x_k \\ &= 2n||x_i||_2^2 - 0. \end{aligned}$$

Therefore

$$\frac{1}{2n} \sum_{k=1}^{n} D_{i,k} + \frac{||D_{i,}||_2^2 - ||D_{k,}||_2^2}{n} = ||x_i||_2^2.$$

We calculate each $||D_{i,}||_2^2$ in time n then the summation over k adds another factor of n. Therefore we can calculate $||x_i||_2^2$ in $O(n^2)$ time.

3. We define a $n \times n$ matrix G where entry $G_{i,j} = x_i^T x_j$. With the set of $||x_i||_2^2$ terms from the previous problem, we can calculate $x_i^T x_j = (||x_i||_2^2 + ||x_j||_2^2 - D_{i,j})/2$ in constant time. So in an additional $O(n^2)$ time we can contruct the matrix G.

   Clearly $G = X^T X$ where X is the $d \times n$ matrix whose columns are the vectors x_i for $i \in [n]$. Then G is positive semi-definite and we can find X either by performing an eigenvalue decomposition or singular value decomposition. The time complexity for both operations is an additional $O(n^3)$.