Massachusetts Institute of Technology
6.854J/18.415J: Advanced Algorithms
David Karger and Aleksander Mądry

Handout 15
Wednesday, October 26, 2016

Problem Set 8

Due: Wednesday, November 2, 2016.

Collaboration policy: collaboration is *strongly encouraged*. However, remember that

1. You must write up your own solutions, independently.
2. You must record the name of every collaborator.
3. You must actually participate in solving all the problems. This is difficult in very large groups, so you should keep your collaboration groups limited to 3 people in a given week.
4. **No bibles. This includes solutions posted to problems in previous years.**

For this PSET, problem (1) is worth 16 points and problem (2) is worth 14 points. The remaining problems are worth the usual 10 points.

Problem 1. Linear System Solving via Gradient Descent.

Suppose we have a linear system $Ax = b$. Here, $A \in \mathbb{R}^{n \times n}$ is assumed to be symmetric, positive semidefinite (PSD), and invertible. Define the condition number of A denoted $\kappa(A)$ as the ratio of its max eigenvalue to its min eigenvalue.

We wish to solve this linear system using gradient descent and will develop a method for doing so in this problem. We will do so by minimizing the function $f(x) = x^\intercal Ax - 2b^\intercal x$ using gradient descent algorithm. In other words, we will start with some guess at a solution $x^{(0)}$ and then generate a sequence of progressively better guesses according to the formula

$$x^{(t)} \leftarrow x^{(t-1)} - \eta \nabla f(x^{(t-1)}).$$

(a) Express the gradient of $f(x) = x^\intercal Ax - 2b^\intercal x$ using a very simple formula that only involves vectors and matrices. Prove that if we find an exact solution to the unconstrained minimization problem $\min x^\intercal Ax - 2b^\intercal x$, this lets us solve $Ax = b$. **Hint:** Recall how one minimizes a function from multivariate calculus. Also, you'll almost certainly need to prove that the function actually has a finite minimum value, rather than $-\infty$.

(b) Define the ℓ_2 norm of a matrix A as $\max_{x \neq \vec{0}} \frac{\|Ax\|_2}{\|x\|_2}$. Prove that if A is symmetric and PSD, its ℓ_2 norm is equal to its maximum eigenvalue. Also convince yourself of the following properties, which you do not need to prove: for any x, $\|Ax\|_2 \leq \|A\|_2\|x\|_2$, and that for any real number $\eta \geq 0$, $\|\eta A\|_2 = \eta\|A\|_2$. **Hints:** Recall that for any symmetric matrix A, the eigenvectors of A form a basis for $\mathbb{R}^n$. Also, note that one of the equivalent definitions of PSD matrices is a symmetric matrix that has nonnegative eigeinvalues.

(c) Define the residual at step t as $r^{(t)} \triangleq Ax^{(t)} - b$. Convince yourself that when x is a solution to $Ax = b$, $\|r^{(t)}\|_2 = 0$ and that when $Ax \neq b$, $\|r^{(t)}\|_2 > 0$. Thus, $\|r^{(t)}\|_2$ measures—in some reasonable sense—how far $x^{(t)}$ is from being a solution to $Ax = b$. (You don't need to write anything for this.)

Now, derive a formula that expresses $r^{(t)}$ solely in terms of $r^{(t-1)}$, A, and η, without any explicit dependence on x. Using this formula, prove that if we set $\eta = \frac{1}{2\|A\|_2}$, then

$$\|r^{(t)}\|_2 \leq (1 - \frac{1}{\kappa(A)})\|r^{(t-1)}\|_2.$$

(d) Conclude a bound on how quickly (in a sense of total runtime) we can get a "solution" $\widetilde{x}$ such that $\|A\widetilde{x} - b\| \leq \epsilon$. You may assume that your algorithm knows good (i.e., accurate up to constant factor) bounds on the max and min eigenvalues of A. **Hint:** You may find it helpful to use—without any proof needed—the fact that for any real number y, $(1 - y) \leq e^{-y}$, which implies that $\prod_i (1 - y_i) \leq e^{-\sum_i y_i}$.

(e) A somewhat more refined notion of an approximate solution to the equation $Ax = b$ can be formulated as follows. Let x^* be the true solution to $Ax = b$. We wish to find a vector x which is close to the true solution x^* in the sense that $\|x - x^*\|_2 \leq \epsilon$. Give a bound on the total runtime required to obtain such solution. **Hint:** Recall that A is invertible.

(f) In this part, you will prove that the whole above analysis can still be carried on even if A is not invertible. Note that in this case there can be *many* solutions to the linear system $Ax = b$.

In order to make this setting sensible, we'll need to change a few definitions and an assumption. To this end, redefine the condition number as the ratio of the maximum eigenvalue over the minimum *nonzero* eigenvalue. Let us also extend the guarantee presented in the previous part to be that we find a vector x that is close to *some* actual solution x^* in the sense that $\|x - x^*\|_2 \leq \epsilon$. Finally, you should assume that a solution actually exists, i.e., that b is in the image of A.

Problem 2. Projected Gradient Descent.

In class, we saw how to use gradient descent to solve unconstrained minimization problems. In this problem, we'll show how it can be used to solve *constrained* minimization problems too.

A *convex set* S is defined as a subset of $\mathbb{R}^n$ such that for every $x, y \in R^n$, every point on the line segment from x to y is also in S; i.e., for all $0 \leq \lambda \leq 1$, the point $\lambda x + (1-\lambda)y \in S$.

Consider the minimization problem $\min f(x)$ s.t. $x \in S$, where f is assumed to be "nice"; ie., it's β-smooth, strongly convex, twice differentiable, and if we differentiate f with respect to some variables, the order of differentiation doesn't change the result. Also, we'll assume S is closed and convex. Finally, we'll assume that there actually does exist a unique optimal solution to this minimization problem.

(a) Prove that the minimization problem $\min f(x)$ s.t. $Ax \geq 0$, is of this form. Thus, the problem we are solving can be used to solve linear programs (among other things).

(b) Define a *contraction operation* as a function $g(x) : \mathbb{R}^n \to \mathbb{R}^n$ that makes points get closer together in the sense that for any $x, y \in \mathbb{R}^n$, we have

$$\|g(x) - g(y)\|_2 \leq \|x - y\|_2.$$

Define the *contraction coefficient* of g as the smallest number $0 \leq w \leq 1$ such that $\|g(x) - g(y)\|_2 \leq w \cdot \|x - y\|_2$ for all $x, y \in \mathbb{R}^n$. Prove that contraction operations compose nicely in the sense that if we have two contraction operations g, h, then $g(h(x))$ is also a contraction operation, and has contraction coefficient at most the product of the contraction coefficients of g and h.

(c) Note that there exists a matrix Z such that $\nabla f(x) - \nabla f(y) = Z(x - y)$ with condition number $\kappa(Z) \leq \kappa(f)$. (Recall that the condition number of a function is defined as the maximum eigenvalue its Hessian can have divided by the minimum eigenvalue it can have.) Then recall the fundamental theorem of line integrals which says that for a "reasonably nice" multivariate function $z(x)$ with one output, the difference of the values the function takes at two points x, y can be expressed as $z(x) - z(y) = \int_{\text{line from } y \text{ to } x} \nabla z(r) \cdot \mathrm{d}\vec{r}$. Apply this theorem using $z(x) = [\nabla f(x)]_i$ for each i gives the result. You don't need to write anything down for this part.

(d) Consider the function g which takes a point, does a gradient descent step, and returns the resulting step, i.e., $g(x) \triangleq x - \eta \nabla f(x)$. Prove that if we set $\eta = \frac{1}{2\beta}$, g is a contraction operation with contraction coefficient at most $(1 - \frac{1}{\kappa(f)})$, where $\kappa(f)$ is the condition number of f.

(e) If we want to use gradient descent to solve $\min f(x)$ s.t. $x \in S$, we could try to do a gradient descent step using f. However, such a step doesn't take into account the constraints and might cause us to violate them. As such, we need to come up with a step that doesn't cause us to violate the constraints. Our step will end up being: do a gradient step (which may cause us to violate the constraints) and then pick the closest point to the resulting point that doesn't violate the constraints.

Let's write this more formally. Let $\Pi_S(x)$ denote the projection of x onto the set S, i.e., $\Pi_S(x)$ returns the vector in S that is closest to x in ℓ_2 distance.

In symbols, $\Pi_S(x) \triangleq \operatorname{argmin}_{y \in S} \|y - x\|_2$. Consider the following update step: $x^{(t)} \leftarrow \Pi_S\left(x^{(t-1)} - \eta \nabla f(x^{(t-1)})\right)$. Prove that $\Pi_S(x)$ is well-defined in the sense that for each x, there is always precisely one point $y \in S$ that is closest to x. You may assume without proof that there is at least one closest point. You just need to show there can't be more than one.

(f) Prove that Π_S is a contraction operation. **Hint:** Use the previous part.

(g) Conclude that after every step, the ℓ_2 distance of x from optimum decreases by a factor of $\left(1 - \frac{1}{\kappa(f)}\right)$ or better when we take the step given in part (e).

(h) Give a bound on the runtime for finding an approximate optimal solution to the minimization problem that has distance at most ϵ from the true optimal solution. Make sure to include the dependence of the runtime on the amount of time it takes to project a vector onto S. (You don't actually need to figure out how long doing the projection takes; just have a variable for it.)

Problem 3. Regularization.

In this problem, we'll develop a technique that can be used to minimize functions which are not strongly convex; ie., with infinite condition number. (As it turns out, this technique is also used in machine learning for the entirely different purpose of preventing overfitting, although we won't examine that angle here.)

Specifically, suppose we wish to solve the unconstrained $\min f(x)$ where $f : \mathbb{R}^n \to \mathbb{R}$ is assumed to be "nice"; ie., it's convex, twice differentiable, and if we differentiate f with respect to some variables, the order of differentiation doesn't change the result. However, f is *not* necessarily strongly convex.

In order to minimize f, we'll turn it into a strongly convex function, then solve the minimize the resulting strongly convex function. Specifically, define, $h(x) = f(x) + \alpha\|x\|_2^2$ where α is some parameter we'll choose later.

(a) Give an example of a convex function that is not strongly convex, but nonetheless has a unique optimal solution. (Strongly convex functions always have unique optimal solutions, but non-strongly convex functions can have them too.)

(b) Prove that the minimum eigenvalue of the sum of a pair of symmetric matrices is at least the sum of their minimum eigenvalues.

(c) Recall from class that like all nice convex functions, the minimum eigenvalue of the Hessian of f is nonnegative. Calculate the Hessian of $\|x\|^2$. Then conclude a lower bound on the strong convexity of $h(x)$.

(d) Prove that if a point y^* minimizes h, it gives an approximately optimal value for f in the sense that for every x^* that minimizes f, we have $f(x^*) \leq f(y^*) \leq f(x^*) + \alpha\|x^*\|_2^2$

(e) Suppose that there exists a distance R such that every point x^* that minimizes f has $\|x^*\| \leq R$. Also suppose that f is β-smooth. Conclude an algorithm for finding a point y^* which is approximately optimal in the sense that the value of f on y^* is within ϵ (additively) of the minimum value of f. Upper bound the runtime as a function of n, R, β, ϵ.

Problem 4. You are given a collection of n points in some metric space (i.e., the distances between the points satisfy the triangle inequality). Consider the problem of dividing the points into k clusters so as to minimize the maximum diameter of (distance between any two points in) a cluster.

(a) Suppose the optimum diameter d is known. Devise a greedy 2-approximation algorithm (an algorithm which gets k clusters each of diameter at most $2d$). **Hint:** consider any point and all points within distance d of it.

(b) Consider the algorithm that (k times) chooses as a "center" the point at maximum distance from all previously chosen centers, then assigns each point to the nearest center. By relating this algorithm to the previous algorithm, show that you get a 2-approximation.

Problem 5. How long did you spend on this problem set? Please answer this question using the Google form that is located on the course website. This problem is mandatory, and thus counts towards your final grade. It is due by the Monday 2:30pm after the pset due date.