The University of Texas at Austin
Department of Electrical and Computer Engineering

EE381K: Large Scale Optimization — Fall 2015

Problem Set Five

Constantine Caramanis — Due: Tuesday, October 13, 2015.

Matlab and Computational Assignments. Please provide a printout of the Matlab code you wrote to generate the solutions to the problems below.

1. **Conjugate Gradient Algorithm.** Recall the linear conjugate gradient algorithm. Download the file `ConjugateGradient.mat`. There you will find matrices and vectors defining two equations: $M_1x = b_1$, and $M_2x = b_2$. The solution, x^*, is there as well, although this is easy to find since both M_1 and M_2 are invertible. Use conjugate gradient to solve these two linear systems, and plot the error, $\log(\|x^{(k)} - x^*\|_{M_i}^2)$ vs. iteration k for both.

2. **BFGS for Quadratic Problems.** In this exercise you use BFGS to solve the same problems you solved by CG above. Solve them by minimizing the corresponding quadratic equation that has the same solution as the linear equation. Note that for non-quadratic problems, as with Newton's method, BFGS should use an approximate line search. In this quadratic case, however, *use an exact line search.* In addition to solving and plotting the error, show that BFGS generates conjugate directions. For quadratic problems, BFGS is also guaranteed to terminate in at most n steps.

3. (?) **Numerical Issues with Conjugate Gradient.** The CG method is most often used for very large problems, because of its computational/complexity advantages, as discussed in class, and demonstrated in the problem above. However, it is also known that CG is more sensitive to numerical instability. Other methods such as Gaussian Elimination are less sensitive to rounding errors than CG.

   Apply the CG algorithm to the system $H\mathbf{x} = \mathbf{b}$, where H is the Hilbert matrix where $H_{i,j} = 1/(i + j - 1)$, $\mathbf{b}$ is the all ones vector, and you initialize from the all zeros vector. Experiment with different accuracies (i.e., terminating when $\|r_k\|$ is larger/smaller), and note the difference in number of steps required.

4. **Newton's Method**. This problem will demonstrate the two convergence behaviors of Newton's method, damped and quadratic, by matlab simulation.

   Consider $f_m(x) = \|x\|^3 + \frac{m}{2}\|x\|^2$ for $m \in \{0, 0.0001, 0.001, 0.1\}$ and $x \in \mathbb{R}^5$.

   (a) For each m, implement Newton's method on $f_m(x)$ and provide the convergence plots, i.e $\log(\|x^{(k)} - x^*\|^2)$ vs. iteration k. Use the constant step size $t = 1$.

   (b) Using the condition for quadratic convergence, explain how and why your result changes according to m.

5. **Central Path**. Consider the linear optimization problem:

$$\begin{aligned} \min: \quad & 2x_1 + 4x_2 + x_3 + x_4 \\ \text{s.t.}: \quad & x_1 + 3x_2 + x_4 \leq 4 \\ & 2x_1 + x_2 \leq 3 \\ & x_2 + 4x_3 + x_4 \leq 3 \\ & x_i \geq 0, \quad i = 1, 2, 3, 4. \end{aligned}$$

(a) Find a function F that is a self-concordant-barrier function, such that the closure of its domain is equal to the feasible set of the problem. (Recall that $-\log(a^\top x - b)$ is a self-concordant-barrier function, as you show in an exercise below.)

(b) Find the analytic center x_F^* using Newton's method. You can initialize at any point in the domain (e.g., $(1/2, 1/2, 1/2, 1/2)$ or any other point you like).

$$x_F^* = \arg \min_{x \in \mathrm{dom} F} F(x).$$

(c) Now you will generate the central path:

$$x^*(t) = \arg \min_{x \in \mathrm{dom} F} f(t; x),$$

where recall:

$$f(t; x) = tc^\top x + F(x).$$

For $t = 0$, the solution, and first point of the central path, is the analytic center. At each iteration, you will compute $t_{k+1} = t_k(1 + \alpha)$. Experiment with different values of α. If α is too small, progress may not be that fast as t will grow slowly. If α is too big, we might move outside the region of quadratic convergence, and although t will grow more quickly, each individual step of the central path will take longer to compute.

(d) Plot the error, $\log(\|x^{(k)} - x^*\|)$ as a function of number of iterations, for different values of α.

Note: Chapter 11 in Boyd & Vandenberghe has much information about central path and barrier methods, although the chapter also contains a lot of information, definitions and ideas we have not yet discussed.

6. Do the same for the (slightly larger) LP contained in `LP_centralpath.mat`. In that file you will find specified: c, A, and b, thus defining the problem:

$$\begin{aligned} \min: \quad & c^\top x \\ \text{s.t.}: \quad & Ax \leq b \\ & x \geq 0. \end{aligned}$$

Note that you can use CVX to quickly solve both this LP and the previous problem, in order to have the solution.

7. **Gradient and Newton**. Consider the function (called the Rosenbrock function)

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

This function is not convex, however it not hard to see that it has a unique minimizer $x^* = (1, 1)$, and that in a neighborhood of this point, the Hessian is positive definite. Initializing at $x_{\text{init}} = (-1.2, 1)$, implement (a) gradient descent and (b) Newton, using a back-tracking line search for both. Plot the error in each as a function of the iteration.

Written Problems

1. (?) We proved quadratic convergence of Newton's method (locally), using the assumption that f is strongly convex, smooth, and with L-Lipschitz Hessian. Locally (so, only after BTLS is taking full steps) derive a rate of convergence in the case where the Hessian is α-Hölder continuous. (Recall that a function F is called α-Hölder continuous if $\|F(x) - F(y)\| \leq H\|x-y\|^\alpha$ for all x, y.)

2. **Self Concordant Barriers.** Recall that a self-concordant function f is called a *self-concordant barrier* if in addition to the properties for self-concordance, it satisfies the property that $\lambda_f(x)^2$ is uniformly bounded by some constant ν. That is, f is called a ν-self-concordant-barrier if
$$\lambda_f(x)^2 = \|\nabla f(x)\|^2_{\nabla^2 f(x)^{-1}} \leq \nu, \qquad x \in \mathrm{dom}(f).$$
   (a) Explain why we need this property (in connection to minimizing a linear function over a convex set).
   (b) For A a positive semidefinite matrix, consider the *concave* quadratic $\phi(x) = c + b^\top x - \frac{1}{2}x^\top Ax$. Show that $f(x) = -\ln\phi(x)$ is a ν-self-concordant barrier function, with $\nu = 1$.

3. **Proof of the Conjugate Gradient Method, part I.** The main thing that remains to be shown is that the conjugate vectors generated by the algorithm are indeed conjugate. We do that in several steps.

   A key property of the proof relates to so-called *Krylov Subspaces*. A Krylov subspace of degree k associated with a matrix A and an initial vector v, is given by $\mathrm{span}\{v, Av, A^2v, \ldots, A^{k-1}v\}$.

   (a) For $\{r_i\}$ the residuals, and $\{p_i\}$ the conjugate vectors generated by the full conjugate gradient algorithm, show that:
$$\begin{aligned}\mathrm{span}\{r_0, r_1, \ldots, r_k\} &= \mathrm{span}\{r_0, Ar_0, \ldots, A^k r_0\}\\ \mathrm{span}\{p_0, p_1, \ldots, p_k\} &= \mathrm{span}\{r_0, Ar_0, \ldots, A^k r_0\}.\end{aligned}$$
   Hint: Proceed by induction. Note that the base case is clear, since $p_0 = -r_0$, and thus the assertion holds for $k = 0$. Then assume that the above holds for a general k, and prove that it holds for $k+1$. Note that $r_{k+1} = Ax_{k+1} - b = A(x_k - \alpha_k p_k) - b = r_k - \alpha_k Ap_k$. This will help you prove the "$\subseteq$" inclusion. For the reverse inclusion, recall that you showed as an in-class exercise that the conjugate vectors are independent.

4. **Proof of the Conjugate Gradient Method, part II.** Now to prove that the conjugate vectors $\{p_i\}$ that are generated by the algorithm are indeed conjugate with respect to A: We proceed by induction. Note that p_0 and p_1 are conjugate by construction of p_1. Therefore, our inductive assumption is that p_k is conjugate to p_i for all $i < k$: $p_k^\top Ap_i = 0$, for all $i = 0, \ldots, k-1$. We must now prove that $p_{k+1}^\top Ap_i = 0$ for all $i = 0, \ldots, k$.

   (a) Show that $r_j \perp r_i$ for all $i < j \leq k+1$. Hint: Note that $r_{i+1} \perp r_i$ holds by construction. Then, proceed by induction on k. To do this, show that $r_{k+1}^\top r_i = r_k^\top r_i - \alpha_k p_k^\top Ar_i$. If $i = k$, this is zero by construction. If $i < k$, the first term is zero by induction. For the second term, use the Krylov results from the previous exercise.
   (b) Again use the Krylov results from the previous exercise, and the orthogonality of the previous part of this exercise, to conclude the main inductive step, namely, that $p_{k+1}^\top Ap_i = 0$ for $i \leq k$.