15.084J Recitation Handout 3

Third Week in a Nutshell:

- Method of Steepest Descent
- Why this method is good
- Why this method is bad
- Line Search Algorithm

Method of Steepest Descent

In order to find a minimum of $f(x)$, take a point x^k, find the direction of steepest descent: $d = -\nabla f(x^1)$. Do a line search in that direction to find the minimum of $f(x^k + \alpha d)$. Call this point x^{k+1}, and repeat until you have an optimum.

Why is this a good idea

- It's easy. Conceptually easy to understand, and really easy to code.
- Actual work in each step is relatively small; it only requires gradients – no hessian computation, no matrix inversion. (Imagine a thousand-dimensional problem; the gradient has a thousand entries, but the hessian has a million!)
- Linear convergence: it's not as fast as Newton, but still quite respectable, and in practice the fact that each iteration is faster might make this a faster way of getting to a solution.
- Always improves from step to step: Newton (without line search) can jump over a solution and make no progress; this is always going to progress from step to step, so you will eventually make progress.
- Fewer requirements: all we need to know is the gradient, so it works even when f is not twice differentiable. In fact, if we have no closed form for the gradient, it's easy to approximate, so we don't even need the gradient to be very nice.

Why is this a bad idea

- The line searching might be expensive; even a log number of steps might be annoying compared to newton without line search.
- When the Hessian is ill-conditioned, it's got really slow linear convergence.

Note that for a quadratic problem, the Hessian is a constant, so the entire problem is "ill-conditioned" or "well-conditioned". For a general problem, the well or ill -conditioned property is local, however it mostly only matters near the optimum (we should rapidly get close, and if the hessian isn't changing too fast, everything near the optimum is equally well or ill conditioned).

An ill conditioned hessian means that the gradient is changing direction rapidly; this causes a problem because our method is only looking at the gradient at one point. Newton's method is solving this problem because it looks at the hessian, and is taking the rate of change of the gradient into account.

Why can't we just check the hessian and find out whether a problem is well conditioned or not, and use Newton if it is ill-conditioned, and steepest descent if it is well-conditioned? Well, you'd find the condition number by finding the hessian and eigenfactoring it. Once you've done that, inverting is trivial, so you've already done the hard part of a Newton step, so this isn't saving you anything.

Bisection Algorithm

How do we find the optimum α for a step of Newton or Steepest Descent? It should be where the derivative wrt α is zero.

We have $\alpha = 0$ is too low; the derivative wrt α at that point is negative.

Go outward in exponentially increasing steps, and in $\log \bar{\alpha}$ time, you'll find a point beyond $\bar{\alpha}$. Then repeatedly bisect the space, and in $\log \bar{\alpha}/\epsilon$, you will reduce the space to size ϵ.

Why use this? It's fast: a log number of steps is good. And again, all we need is to take a derivative at each step. If taking second derivatives is easy, you can use newton (it's just a minimization problem in one dimension, and we don't even need to invert a matrix).