

Recitation notes

April 11, 2008

For written homeworks, electronic submission is much preferred. You might want to check out lyx (<http://www.lyx.org/>) or texmacs (<http://www.texmacs.org/>).

1 Boyd and Vandenberghe, Section 9.1.2

1.1 $f(x) - p^* \leq \frac{1}{2m} \|\nabla f(x)\|^2$ (section 9.1.2)

If f is strongly convex, then:

$$\nabla^2 f(x) \succeq mI$$

For any y , if we let $g(\lambda) = f(\lambda y + (1 - \lambda)x)$, then by Taylor's theorem, there exists a $\lambda_0 \in [0, 1]$ such that:

$$\begin{aligned} g(1) &= g(0) + g'(0) + \frac{1}{2}g''(\lambda_0) \\ f(y) &= f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(\lambda y_0 + (1 - \lambda_0)x)(y - x) \end{aligned}$$

What this has shown is that, from Taylor's theorem in one dimension along the line defined by x and y , we may easily derive the more general multidimensional version that there exists a $z = \lambda y_0 + (1 - \lambda_0)x$ with $\lambda_0 \in [0, 1]$ such that:

$$f(y) = f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(z)(y - x)$$

Now, by strong convexity, $\nabla^2 f(z) \succeq mI$, so that $w^T \nabla^2 f(z) w \geq m$ for all w such that $\|w\| = 1$, so that $(y - x)^T \nabla^2 f(z)(y - x) \geq m \|y - x\|_2^2$.

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{m}{2} \|y - x\|_2^2 \quad (1)$$

Note that the RHS above is a convex function of y , so that it is minimized if its gradient with respect to y is zero:

$$\begin{aligned} h(y) &= f(x) + \nabla f(x)^T (y - x) + \frac{m}{2} \|y - x\|_2^2 \\ \nabla h(y) &= \nabla f(x) + m(y - x) \end{aligned}$$

Setting $\nabla h(y) = 0$:

$$y = x - \frac{1}{m} \nabla f(x)$$

So that, substituting back into the original inequality 1:

$$\begin{aligned} f(y) &\geq f(x) - \frac{1}{m} \|\nabla f(x)\|_2^2 + \frac{1}{2m} \|\nabla f(x)\|_2^2 \\ &\geq f(x) - \frac{1}{2m} \|\nabla f(x)\|_2^2 \end{aligned}$$

The above holding for any y . In particular, it holds for $y = x^*$, for which $f(y) = p^*$:

$$p^* \geq f(x) - \frac{1}{2m} \|\nabla f(x)\|_2^2$$

Rearranging terms:

$$f(x) - p^* \leq \frac{1}{2m} \|\nabla f(x)\|_2^2$$

So that, in words, if the gradient is small at some x , then x is close to being optimal.

1.2 $\|x - x^*\|_2 \leq 2m \|\nabla f(x)\|_2$

Return to inequality 1:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{m}{2} \|y - x\|_2^2$$

Let $y = x^*$ again:

$$p^* \geq f(x) + \nabla f(x)^T (x^* - x) + \frac{m}{2} \|x^* - x\|_2^2$$

By Cauchy-Schwarz, $\nabla f(x)^T (x^* - x) \geq -\|\nabla f(x)\|_2 \|x^* - x\|_2$:

$$\begin{aligned} p^* &\geq f(x) - \|\nabla f(x)\|_2 \|x^* - x\|_2 + \frac{m}{2} \|x^* - x\|_2^2 \\ f(x) - p^* &\leq \|\nabla f(x)\|_2 \|x^* - x\|_2 - \frac{m}{2} \|x^* - x\|_2^2 \end{aligned}$$

And since $f(x) - p^* \geq 0$ by definition:

$$\begin{aligned} 0 &\leq \|\nabla f(x)\|_2 \|x^* - x\|_2 - \frac{m}{2} \|x^* - x\|_2^2 \\ \|x^* - x\|_2 &\leq \frac{2}{m} \|\nabla f(x)\|_2 \end{aligned}$$

So we see that if the gradient is small at some x , then not only is $f(x)$ close to $f(x^*) = p^*$ (which we already showed), but x is close to x^* .

2 Steepest descent with l^1 norm (Boyd and Vandenberghe, section 9.4.2)

In steepest descent under the l^1 norm, we find, at each iteration, a search direction which satisfies:

$$\Delta x = \arg \min_v \left\{ \nabla f(x)^T v \mid \|v\|_1 \leq 1 \right\}$$

Note that by Hölder's inequality, and using the fact that $\|v\|_1 \leq 1$:

$$\begin{aligned} \sum_{i=1}^n |(\nabla f(x))_i v_i| &\leq \|\nabla f(x)\|_\infty \|v\|_1 \\ &\leq \|\nabla f(x)\|_\infty \end{aligned}$$

Since $\nabla f(x)^T v \geq -\sum_{i=1}^n |(\nabla f(x))_i v_i|$, and negating the above inequality:

$$\nabla f(x)^T v \geq -\|\nabla f(x)\|_\infty$$

Let i be an index which maximizes $|(\nabla f(x))_i|$. Then if we let $\Delta x = -\text{sgn}((\nabla f(x))_i) e_i$ (the i th standard unit basis vector), we'll have that:

$$\nabla f(x)^T \Delta x = -|(\nabla f(x))_i|$$

And since, by assumption, $|(\nabla f(x))_i| = \|\nabla f(x)\|_\infty = \max_i |(\nabla f(x))_i|$, we see that this choice of Δx satisfies the arg min defining it.

Hence, in steepest descent with the l^1 norm, we will always “move” along a single coordinate of the vector x , namely that which has the largest corresponding gradient component. This is called “coordinate descent”

3 Affine invariance of Newton's method (Boyd and Vandenberghe, section 9.5.1)

Suppose that $A \in \mathbb{R}^{n \times n}$ is nonsingular, and $b \in \mathbb{R}^n$. Define:

$$g(x) = f(Ax + b)$$

Then:

$$\begin{aligned} \nabla g(x) &= A^T \nabla f(Ax + b) \\ \nabla^2 g(x) &= A^T \nabla^2 f(Ax + b) A \end{aligned}$$

So that the Newton step for g will be:

$$\begin{aligned} \Delta x &= -(\nabla^2 g(x))^{-1} \nabla g(x) \\ &= -A^{-1} (\nabla^2 f(Ax + b))^{-1} A^{-T} A^T \nabla f(Ax + b) \\ &= -A^{-1} (\nabla^2 f(Ax + b))^{-1} \nabla f(Ax + b) \end{aligned}$$

Where $-\left(\nabla^2 f(Ax + b)\right)^{-1} \nabla f(Ax + b)$ is the Newton step for f at $Ax + b$. Therefore, if we let $y = Ax + b$, and $\Delta y = -\left(\nabla^2 f(Ax + b)\right)^{-1} \nabla f(Ax + b)$:

$$y + \Delta y = Ax + b + A\Delta y$$

That is, Newton's method for f will follow an affine-transformed version of the path followed by Newton's method for g , where this affine transformation is exactly that which maps x to y . In other words, Newton's method is Affine invariant.

4 Boyd and Vandenberghe, section 9.7

4.1 Precomputation of line searches

Suppose that $f(x) = \phi(Ax + b)$, where $A \in \mathbb{R}^{p \times n}$ and $\phi: \mathbb{R}^p \rightarrow \mathbb{R}$ is "easy" to compute, so that the main "effort" involved in computing $f(x)$ is in computing Ax . When we're performing the line search, however, we'll always be looking along the same "direction" (Δx), so that precomputing $A\Delta x$ can speed things up.

Note that the cost of computing $Ax + b$ directly is $2pn$ floating point operations (pn multiplies, and pn adds). If we let $b' = Ax + b$ and $a' = A\Delta x$ (the first of which costs $2pn$ operations, and the second $(2p - 1)n$), then we may compute $A(x + t\Delta x) + b$ for any t as:

$$\begin{aligned} A(x + t\Delta x) + b &= tA\Delta x + Ax + b \\ &= ta' + b' \end{aligned}$$

Which costs $2pk$ operations. Hence, if, during the line search, we compute $A(x + t\Delta x) + b$ for k different values of t , then computing each value directly would cost $2pnk$ operations, while precomputing a' and b' , and then using these quantities to speed up the line search, would cost $(2pn + (2p - 1)n) + 2pk \leq 4pn + 2pk$ operations.