CS 224: Advanced Algorithms Fall 2014

Lecture 17 — October 28, 2014

Prof. Jelani Nelson *Scribe: Ben Thompson*

1 Overview

Today we cover interior point methods. This is a method for solving convex programs in general, but, for the case of Linear Programs (LPs), was introduced by Karmarkar [Kar84].

We consider the standard formulation of an LP:

$$\begin{aligned} &\min c^T x \\ &s.t. \quad Ax \geq b \end{aligned}$$

where $A \in \mathbb{R}^{m \times n}$ and $x, c \in \mathbb{R}^n, b \in \mathbb{R}^m$. Define the feasible region $P = \{x : Ax \geq b\}$.

The basic idea of an interior point method (IPM) is to minimize a function, $f_\lambda(x) = \lambda c^T x + p(s(x))$, where $s(x)_i$ is the slack $\langle a_i, x \rangle - b_i$ and $p(s(x)) \to \infty$ if any $s(x)_i \to 0$.

The function p is called a "barrier" function and discourages the solution from approaching the boundaries of the region P. To be more concrete, we use $p(x) = \sum_{i=1}^m \log(s(x)_i)$ as our barrier function. Note, that this approaches infinity as slack goes to zero (as point goes to boundary).

Consider the two limiting cases As $\lambda \to 0$, minimization of the original LP is negligible. As a result, the minimum of $f_\lambda(x)$ moves towards the "analytic" center of the feasible region, P. As $\lambda \to \infty$, $x(\lambda) \to x^*$ where x^* is the solution to the original LP. Hence, our plan is to develop an algorithm that starts from a solution of the optimization problem for very small λ and then *iteratively* increases λ until it is sufficiently large that the solution to the minimization of $f_\lambda(x)$ is a very good approximation of the minimizer of the original LP.

We have now converted our LP into the minimization of the function f_λ.

Outline of algorithm (Steps 1 through 4 are intialization. Step 5 is the meat of the iterative algorithm. In this lecture, we only discuss step 5 in detail).

1. Modify the original, LP, to LP', such that LP' has some trivial interior point.
2. We will show that if the LP is bounded and feasible, then the solution to the modified LP' can easily be converted to a solution of the original LP.
3. Suppose we know a point x is an initial interior point of LP'.
4. From x, obtain a $\tilde{x}(\lambda_0)$ which approximately minimizes $f_{\lambda_0}(x)$.
5. See next section!

Step 5

The general idea for step 5 is to choose a λ_{k+1} close enough to (but larger than!) λ_k such that $\tilde{x}(\lambda_k)$ is close enough to $\tilde{x}(\lambda_{k+1}$ that an iterative optimization can be used to quickly update our solution. Eventually, λ will be large enough that the approximate minimizer of f will be close enough to the optimal solution of the LP.

A few questions remain:

- How small does λ_0 need to be? Define $L \approx 1 + \max\{\log(\|b\|_\infty), \log(\|c\|_\infty), \log(detmax(A))\}$ where $detmax(A)$ is the largest determinant of any square submatrix of A. Then $\lambda_0 = e^{-cL}$ for some c. In other words, λ_0 must be exponentially small in terms of the input.
- How small does the error need to be before stopping? We stop when the error is exponentially small or $c^T x(\lambda) - c^T x^* < \epsilon (= e^{-cL})$. When the error in the solution of the original LP is small enough, we can round to a vertex for the true minimum.
- How large does λ_k need to be before stopping? We need to translate the error criterion above into a constraint on λ. Observation: $f_\lambda(x)$ is strictly convex from R^n to $\mathbb{R}$, so there is a unique minimizer, x_λ and the gradient $\nabla f_\lambda(x) = 0$. Given the barrier function $p(s(x))$ given earlier, the gradient: $\nabla f_\lambda(x) = \lambda c - A^T S_x^{-1}\mathbf{1}$, where $\mathbf{1}$ is the all ones vectorand S_x is the matrix with each slack on the diagonal. Hence,

  $$0 = \langle 0, x(\lambda) - x^* \rangle = \langle \nabla f, x(\lambda) - x^* \rangle = \langle \lambda c - A^T S_x^{-1}\mathbf{1}, x(\lambda) - x^* \rangle \tag{1}$$

  which implies

  $$\lambda c^T (x(\lambda) - x^*) = \mathbf{1}^T S_x^{-1} A(x(\lambda) - x^*) = \mathbf{1}^T S_x^{-1}(s(x(\lambda)) - s(x^*)) = \sum_{i=1}^{m} \frac{s(x(\lambda)) - s(x^*)}{s(x(\lambda))} \leq m \tag{2}$$

  Rearranging:

  $$c^T x(\lambda) - c^T x^* \leq \frac{m}{\lambda} \tag{3}$$

  As a result, once $\lambda \geq m/\epsilon \approx m e^{cL}$, the minimizer of $f_\lambda(x)$ is a good enough approximation to minimizer of the original LP that we can just round to a vertex.
- The final question is what factor to increase λ each iteration. We will show that we can take $\lambda_{k+1} = (1 + \frac{1}{\sqrt{m}})\lambda_k$. Given this step ratio, $O(\sqrt{m}(L + \lg m))$ are sufficient for convergence.

In comparison, the original IPM by Karmarkar[Kar84] required $O(mL)$ iterations. This was improved to $O(\sqrt{m}L)$ iterations by Renegar [Ren88]. Recently, this was improved to $\widetilde{O}(\sqrt{rank(A)})$ by Lee and Sidford [LS13].

The vague idea we have outlined is called a "path following" IPM. Define $(\lambda, x(\lambda))\forall\lambda \geq 0$. This is a curve called the "central path". Path following IPMs start somewhere near the beginning of the curve (near the analytic center) and try to follow the curve closely. Due to the approximation minimization of $f(\lambda)$, the IPM can't follow the central path exactly, instead following $(\lambda, \tilde{x}(\lambda))$.

Optimization

Once we have a solution that is somewhat accurate, how do we improve the error so that the solution is an approximate minimizer? We use iterative methods for computing $x^* = \text{argmin}_{x \in \mathbb{R}^n} f(x)$ where $f : \mathbb{R}^n \to \mathbb{R}$.

An iterative methods starts with $x_0 \in \mathbb{R}^n$ and iteratively compute x_{k+1} from x_k using some rule that improves the error at each step.

First order method – Gradient Descent

We assume:

1. There is some oracle that given $x \in \mathbb{R}^n$, tells us $f(x)$ and $\nabla f(x)$.
2. Some bound is given to us on how the Hessian matrix, $\nabla^2 f$, can behave. In particular $\exists\ 0 < \mu < L$ such that $\mu I \preceq \nabla^2 f(x) \preceq LI \quad \forall x \in \mathbb{R}^n$. The meaning of $A \preceq B$: $A - B$ is positive semi definite (PSD), or that $\forall x \quad x^T(B - A)x \geq 0$ (called a "Loewner ordering"). Remember $\nabla^2 f(x) \in \mathbb{R}^{nxn}$ where $\nabla^2 f(x)_i j = \frac{\partial^2}{\partial x_i \partial x_j} f(x)$. Essentially, this bound limits the eigenvalues of the Hessian matrix to within a certain range. In other words, the matrix is well-conditioned.

Now, we would like an update rule such that $f(x_{k+1}) - f(x^*) \leq \alpha(f(x_k) - f(x^*))$, with $\alpha < 1$. One such rule is gradient descent. Given some iterate x_0, gradient descent updates to $x_1 = x_0 - \frac{1}{L}\nabla f(x_0)$

To analyze this rule, using Taylor's theorem:

$$\exists\ \hat{x}_\alpha \quad s.t. \tag{4}$$

$$f(x_1) = f(x_0) + \langle \nabla f(x_0), x_1 - x_0 \rangle + E \tag{5}$$

$$\text{where} \quad E = \int_0^1 \int_0^t \langle x_1 - x_0, \nabla^2 f(\hat{x}_\alpha)(x_1 - x_0) \rangle dx dt \int\int \leq L\|x_1 - x_0\|_2^2. \tag{6}$$

So, in general:

$$f(x_1) \leq f(x_0) + < \nabla f(x_0), x_1 - x_0 > + \frac{L}{2}\|x_1 - x_0\|^2 \tag{7}$$

The gradient descent rule, $x_1 = x_0 - \frac{1}{L}\nabla f(x_0)$, minimizes the right hand side of 7. Concretely:

$$f(x_1) \leq f(x_0) - \frac{1}{2L}\|\nabla f(x_0)\|_2^2 \tag{8}$$

Now, we need to use the lower bound on the Hessian matrix from 2 above to transform this into a convergence statement:

$$f(y) \geq f(x) + < \nabla f(x), y - x > -\mu/2\|x - y\|_2^2 \quad \forall (x, y) \tag{9}$$

With this bound, Fix $x = x_k$ and take the minimum over all y. The minimal value of $f(y)$ is clearly $f(x^*)$ (definition!).

$$f(x^*) \geq \min_y \left(f(x_k) + < \nabla f(x_k), y - x_k > - \frac{\mu}{2} \|x_k - y\|_2^2 \right) \tag{10}$$

$$= f(x_k) - \frac{1}{2\mu} \|\nabla f(x_k)\|_2^2 \quad \forall (x_k, y) \tag{11}$$

$$\implies \|\nabla f(x_k)\|_2^2 \geq 2\mu (f(x_k) - f(x^*)) \tag{12}$$

Combining equations 8 and 12, we get:

$$f(x_{k+1}) - f(x^*) \leq f(x_k) - f(x^*) - \frac{1}{2L} \|\nabla f(x_0)\|_2^2 \tag{13}$$

$$\leq (1 - \mu/L) f(x_k) - f(x^*). \tag{14}$$

Thus $\alpha = 1 - \mu/L$. As a result, each step converges with a rate dependent on the ratio of the eigenvalues of the Hessian matrix.

To guarantee that we cut the error in half, gradient descent need $O(\frac{L}{\mu})$ iterations. Further, to reduce from Δ to $\epsilon\Delta$ requires $O(\frac{L}{\mu} \log \epsilon)$ iterations. This is not optimal. Another method called "Accellerated gradient descent" cuts the error in half in $O(\sqrt{\frac{L}{\mu}})$ iterations [Nes83]

Next lecture, we will describe a second order optimization method – Newton's method – and bounds on its convergence rates. Then, Newton's method will be used to get $\tilde{x}(\lambda_{k+1})$ from $\tilde{x}(\lambda_k)$, finishing step 5 of the IPM algorithm.

References

[Kar84] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–396, 1984.

[LS13] Yin Tat Lee and Aaron Sidford. Matching the universal barrier without paying the costs : Solving linear programs with $\tilde{O}(\sqrt{rank})$ linear system solves. *CoRR*, abs/1312.6677, 2013.

[Nes83] Yurii Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[Ren88] James Renegar. A polynomial-time algorithm, based on Newton's method, for linear programming. *Mathematical Programming*, 40(1–3):59–93, 1988.