CS 270: Combinatorial Algorithms and Data Structures Spring 2023

Lecture LEC 23 — 2023-04-11

Prof. Jelani Nelson *Scribe: Johannes Pfannschmidt*

1 Overview

In the last lecture we covered strong duality, complementary slackness, and the ellipsoid algorithm.

This lecture's topic is the interior point method (IPM) for linear programming. This includes a look at newton's method, with a detour to gradient descent.

2 IPM

2.1 Input

The interior point method takes as input an LP of the form:

$$\min_x c^T x$$
$$\text{s.t. } Ax \geq b$$

2.2 Basic Idea

The idea is to define a series of functions:

$$\begin{aligned} \lambda &\in [0,\infty) \\ f_\lambda(x) &:= \lambda c^T x + p(s(x)) \\ \text{where } p \text{ is s.t. } p(z) &\to \infty \\ \text{if any } z_i &\to 0 \end{aligned}$$

$s(x) := Ax - b$ is called the slack-vector, and p is called the barrier function, as it keeps the point interior to the LP by penalization.

For this lecture, $p(s(x)) := -\sum_{i=1}^m ln(s(x)_i)$.

For IPM, we start at the optimal x for $\lambda = 0$, and then gradually increase λ while continuously adapting x to stay optimal for the current λ. In doing so, we move from the polytope's analytical center to the optimal vertex along the unique central path, as visualized in Fig. 1.

Since the algorithm is actually iterative and moves in discrete steps, we in reality don't move exactly along the central path, but stay ”close” to it. We differentiate levels of ”closeness”:

- **central**: x is on central path (and gradient is 0)
- **awesome**: Norm of gradient is is tiny ($\leq \frac{1}{100}$)
- **ok**: Norm of gradient is small ($\leq \frac{1}{3}$)

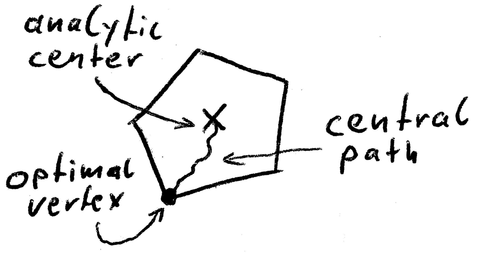

Figure 1: Rough visualization of the IPM method

2.3 Overall architecture

The algorithm operates as follows:

1. set λ_0 to be very small: $\lambda_0 = \exp(-cL)$
2. get $\tilde{x}(\lambda_0)$ that is awesome
3. for $i = 1..k$
   - $\lambda_i = (1+\gamma)\lambda_{i-1}$
   - run $O(1)$ Newton iterations on $\tilde{x}(\lambda_{i-1})$ for f_{λ_i} to obtain $\tilde{x}(\lambda_i)$ that is awesome

(L is the precision of the problem.)

For this to work, γ needs to be sufficiently small such that if $\tilde{x}(\lambda_{i-1})$ is awesome, $x(\lambda_i)$ is still at least ok. Values $\gamma = O(\frac{1}{\sqrt{m}})$ and $k = O(\sqrt{m}L)$ will work.

2.4 Finding $\tilde{x}(\lambda_0)$ in step 2.

To find our starting point, we pick an N to be really big ($\exp(cL)$), and formulate a new LP:

$$\begin{aligned} &\min_x c^T x + Nz \\ \text{s.t.} \quad &Ax + z \cdot \mathbb{1} \geq b \\ &0 \leq z \leq 2^{L+1} \\ &-2^{L+1}\mathbb{1} \leq x \leq 2^{L+1}\mathbb{1} \end{aligned}$$

This LP has an easy interior point: $z = ||b||_0, x = \vec{0}$.

Using

$$\nabla f_\lambda(x) = \lambda c - A^T S_x^{-1}\mathbb{1} \tag{1}$$

$$\nabla^2 f_\lambda(x) = A^T S_x^{-2} A \tag{2}$$

and $||x||_A = \sqrt{x^T Ax}$ for PSD x, we define "centrality" for λ as $\delta_\lambda(x) := ||\nabla f_\lambda(x)||_{(\nabla^2 f_\lambda(x))^{-1}}$.

This is also the norm we use for above mentioned definition of "awesome" ($\delta_\lambda(x) \leq \frac{1}{100}$) and "ok" ($\delta_\lambda(x) \leq \frac{1}{3}$).

To now find our $\tilde{x}(\lambda_0)$ for step 2., we first set $\lambda = 1$, and introduce a different cost function $c' = A^T S^{-1}_{(x_0,z_0)}\mathbb{1}$, such that the gradient is $1c' - A^T S^{-1}_{(x_0,z_0)}\mathbb{1} = 0$, giving us a point on the central path.

To obtain a starting point on the central path for our original cost function, we apply step 3. in reverse, decreasing λ from one towards zero. Since for sufficiently small λ the cost function does not influence the gradient much, we can then switch back to the old cost function, and our point is still "awesomely close" to the central path. We have found our $\tilde{x}(\lambda_0)$.

2.5 3rd step

The question remains of how to decide how large k in the 3rd step should be:

$$
\begin{aligned}
\text{let } x(\lambda) := \min_x f_\lambda(x)& \\
0 &=< 0, x(\lambda) - x^* > \\
&=< \lambda c - A^T S_x^{-1}(\lambda)\mathbb{1}, x(\lambda) - x^* > \\
&\Rightarrow \lambda c^T(x(\lambda) - x^*) = \mathbb{1}^T S_x^{-1}(\lambda) A_{(x(\lambda)-x^*)} \\
&= \mathbb{1}^T S^{-1}_{x(\lambda)}(s(x(\lambda) - x^*)) \\
&= \sum_{i=1}^{m} \frac{s(x(\lambda))_i - s(x^*)_i}{s(x(\lambda))_i} \\
&\leq= \sum_{i=1}^{m} \frac{s(x(\lambda))_i}{s(x(\lambda))_i} = m \\
&\Rightarrow c^T(x(\lambda) - x^*) \leq \frac{m}{\lambda}
\end{aligned}
$$

where x^* is the perfect (central) x for λ.

If the error $c^T(x(\lambda) - x^*)$ should be at most ϵ, we therefore need λ to be $\frac{m}{\epsilon}$.

This is the termination criterion for the 3rd step, we stop iterating if $\lambda \geq \frac{m}{\epsilon}$. Actually, if $\frac{m}{\lambda} < \exp(-cL)$, we can round to the optimal vertex (proof omitted). Therefore we are done when $\frac{m}{\lambda} < exp(-cL) \Rightarrow \lambda > m \cdot \exp(cL)$.

Specifically, to be done we need $(1+\gamma)^k \lambda_0 > m \cdot \exp(cL)$ to hold, or derived therefrom $(1+\gamma)^k > m \cdot \exp(2cL)$, which for the number of iterations implies $k \geq \frac{1}{\gamma} \cdot \ln(m \cdot \exp(2cL)) = O(\sqrt{m}(L + \lg m))$

3 Detour into continuous optimization

3.1 First-order methods

- given $f : \mathbb{R}^n \to \mathbb{R}$, ∇f
- promised $\forall x : \alpha I \leq \nabla^2 f(x) \leq \beta I$ [1]

3.1.1 Basic Idea

- start with some iterate $x_0 \in \mathbb{R}^n$
- gradually move from x_k to x_{k+1} along negative gradient, such that $f(x_{k+1}) < f(x_k)$

This is motivated by Taylor's theorem:

$$\begin{aligned} f(x_{k+1}) &= f(x_k) + < \nabla f(x_k), x_{k+1} - x_k > \\ &\qquad + \int_0^1 \int_0^t < x_{k+1} - x_k, \nabla^2 f(x_\alpha)(x_{k+1} - x_k) > d\alpha \; dt \\ &\qquad \text{where } x_\alpha := x_k + \alpha(x_{k+1} - x_k) \text{ for } x \in [0,1] \\ &\leq f(x_k) + < \nabla f(x_k), x_{k+1} - x_k > + \frac{\beta}{2} ||x_{k+1} - x_k||_2^2 \end{aligned}$$

Gradient descent then essentially is choosing x_{k+1} to minimize the last line directly above, which gives $x_{k+1} = x_k - \frac{1}{\beta} \nabla f(x_k)$. With that, said line will be at most $f(x_k) - \frac{1}{2\beta} ||\nabla f(x_k)||_2^2$ (not shown).

3.2 Lemma (proof omitted)

$$\forall k : f(x_{k+1}) - f(x^*) \leq (1 - \frac{\alpha}{\beta}) \cdot (f(x_k) - f(x^*))$$

$\Rightarrow$ The optimality gap can be halved in $O(\frac{\beta}{\alpha})$ oracle calls.

Notably, a better bound of $O(\sqrt{\frac{\beta}{\alpha}})$ is achievable using "accelerated gradient descent", due to Nesterov [1] in '83. This bound is optimal.

4 Newton's Method

- goal: $\min f(x)$
- given: f, ∇f, $\nabla^2 f$

[1] Notation: $A \leq B \Leftrightarrow B - A$ is PSD $\Leftrightarrow \forall z : z^T A z \leq z^T B z$. This is called the Loewner order.

- **Assumption**:

$$
\begin{aligned}
&\forall k, \quad x_\alpha := x_k + \alpha(x_{k+1} - x_k) : \\
&(1-\epsilon)\nabla^2 f(x_k) \leq \nabla^2 f(x_\alpha) \leq (1+\epsilon)\nabla^2 f(x_k) \\
&\Leftrightarrow \\
&-\epsilon I \leq A^{\frac{1}{2}}(B-A)A^{\frac{1}{2}} \leq \varepsilon I
\end{aligned}
$$

4.1 Heart of Newton

In Newton's method, the function f is approximated by its second-order Taylor expansion:

$$f(x_{k+1}) \approx f(x_k) + < \nabla f(x_k), x_{k+1} - x_k > + \frac{1}{2} < x_{k+1} - x_k >, \nabla^2 f(x_k)(x_{k+1} - x_k) >$$

x_{k+1} is then choosen to optimize the right-hand side of this equation, which gives:

$$x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1} \cdot \nabla f(x_k)$$

4.1.1 Newton Progress Lemma

If f is twice differentiable and **Assumption** holds, then:

$$||\nabla f(x_{k+1})||_{(\nabla^2 f(x_{k+1}))^{-1}} \leq \frac{\epsilon}{1-\epsilon} ||\nabla f(x_k)||_{(\nabla^2 f(x_k))^{-1}}$$

References

[1] Nesterov Y., A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, volume 27, pages 137–147, 1983.