IE 598: Big Data Optimization **Fall 2016**

Lecture 7: Introduction to Optimization Algorithms – September 13

Lecturer: Niao He *Scriber: Shripad Gade*

Overview: In this lecture we conclude the discussion on Conic Programming. We revisited the definitions of $\mathcal{K}$ representable functions (and sets), and discussed its applications in rewriting convex optimization problems into conic programs. We also discussed duality in conic programs followed by a brief introduction to Interior Point Methods. We started a discussion on taxonomy for optimization algorithms along with the notion of error in optimization.

7.1 Conic Programming

7.1.1 Recap

Given a cone $\mathbf{K}$ in $\mathbb{R}^m$ (convex, pointed, closed with nonempty interior), an objective $c \in \mathbb{R}^n$, a $m \times n$ constraint matrix A with right hand side $b \in \mathbb{R}^m$, a conic program is defined as -

$$\min_x \{c^T x \mid Ax - b \geq_{\mathbf{K}} 0\}. \tag{7.1}$$

A few special types of conic programs -

- (Non-negative Orthant) $\mathbf{K} = \mathbb{R}^m_+$, the constraint will become $Ax - b \geq 0$. The resulting conic program is a Linear Program (LP).
- (Lorentz Cone) $\mathbf{K} = \mathcal{L}^m$, the constraint can be written in the form of $\|Ax - b\|_2 \leq c^T x - d$. This gives a Second Order Cone Program (SOCP), also called Conic Quadratic Program (CQP).
- (Semi-definite Cone) $\mathbf{K} = S^m_+$, the constraint is $\sum x_i A_i - b \succcurlyeq 0$. The resulting program is a Semi-definite Program (SDP).

As proved in Lecture 6, LP $\subseteq$ SOCP $\subseteq$ SDP.

Let $\mathcal{K}$ be a family of regular cones closed with taking direct product.

Definition 7.1 ($\mathcal{K}$ Representable Sets) *A set X is called conic representable or $\mathcal{K} - r$ if $\exists\ \boldsymbol{K} \in \mathcal{K}$ such that, $X = \{x : \exists u, Ax + Bu - b \in \boldsymbol{K}\}$.*

Definition 7.2 ($\mathcal{K}$ Representable Functions) *A function $f(x)$ is $\mathcal{K} - r$ if $epi(f) = \{(x,t) : t \geq f(x)\}$ is a $\mathcal{K} - r$ set, i.e. $\exists\ \boldsymbol{K} \in \mathcal{K}$ such that, $epi(f) = \{(x,t) : \exists v, Cx + dt + Dv - e \in \boldsymbol{K}\}$*

Using the above definitions we can claim that if $f(x)$ is $\mathcal{K} - r$ function and X is a $\mathcal{K} - r$ set, then any optimization programs can be equivalently written as,

$$\begin{array}{lll} \min\limits_{x} & f(x) & \\ s.t. & x \in X & \end{array} \iff \begin{array}{lll} \min\limits_{x,t} & t & \\ s.t. & t \geq f(x) & \\ & x \in X & \end{array} \iff \begin{array}{lll} \min\limits_{x,t,u,v} & t & \\ s.t. & Cx + dt + D\nu - e \in \mathbf{K}_f & \\ & Ax + Bu - b \in \mathbf{K}_X & \end{array}$$

Examples of SOCP-r functions

- $f(x) = \|x\|_2$, we have $epi(f) = \{(x,t) : t \geq \|x\|_2\} \iff \{(x,t) : \begin{bmatrix} x \\ t \end{bmatrix} \in \mathcal{L}^{n+1}\}$
- $f(x) = x^T x$, we have $epi(f) = \{(x,t) : t \geq x^T x\} \iff \begin{bmatrix} 2x \\ t-1 \\ t+1 \end{bmatrix} \in \mathcal{L}^{n+2}$
- $f(x) = x^T Q x + q^T x + r$ where $Q = LL^T \succcurlyeq 0$

  $epi(f) = \{(x,t) : t \geq x^T Q x + q^T x + r\} \iff \left\{ (x,t) : \begin{bmatrix} 2L^T x \\ t - q^T x - r - 1 \\ t - q^T x - r + 1 \end{bmatrix} \in \mathcal{L}^{n+2} \right\}$
- $f(x) = x^2 + 2x^4$, we have

$$epi(f) = \left\{ (x,t) : \exists\, t_1, t_2 \text{ s.t. } \begin{bmatrix} 2x \\ t_1 - 1 \\ t_1 + 1 \end{bmatrix} \in \mathcal{L}^3, \begin{bmatrix} 2t_1 \\ t_2 - 1 \\ t_2 + 1 \end{bmatrix} \in \mathcal{L}^3, \begin{bmatrix} 0 \\ t - t_1 - 2t_2 \end{bmatrix} \in \mathcal{L}^2 \right\}$$

Examples of SDP-r functions

- $f(X) = \lambda_{\max}(X)$ on S_+^m.
  Note, $t \geq \lambda_{\max}(X) \iff tI_m - X \succcurlyeq 0$.
- $f(X) = \sigma_{\max}(X) = \max_{\|u\|_2 \leq 1} \|Xu\|_2$ on $\mathbb{R}^{m \times n}$.
  Note, $t \geq \sigma_{\max}(X) \iff t \geq \lambda_{\max}(X^T X) \iff t^2 I - X^T X \succcurlyeq 0 \underset{Schur}{\iff} \begin{bmatrix} tI_n & X^T \\ X & tI_m \end{bmatrix} \succcurlyeq 0.$

Example - [Group Lasso] We can show that sparse group lasso

$$\min_w \left\{ \|Xw - y\|_2^2 + \lambda \sum_{i=1}^{p} \|w_i\|_2 \right\}$$

is a SOCP problem. We can reformulate this problem into a SOCP as follows -

$$\min \quad t_0 + \lambda \sum_{i=1}^{k} t_i \tag{7.2}$$

$$s.t. \quad \begin{bmatrix} 2(Xw - y) \\ t_0 - 1 \\ t_0 + 1 \end{bmatrix} \in \mathcal{L}^{m+2}, \ \begin{bmatrix} w_i \\ t_i \end{bmatrix} \in \mathcal{L}^{n_i+1} \ \forall i \tag{7.3}$$

7.1.2 Conic Problem - Duality

Duality in conic programs is simple. We can write a conic program and its dual as follows:

$$\text{CP (P)}: \quad \min_x \{c^T x | Ax - b \succcurlyeq_{\mathbf{K}} 0\}$$

$$\text{CP Dual (D)}: \quad \max_y \{b^T y | A^T y = c, y \succcurlyeq_{\mathbf{K}_*} 0\}$$

The dual program is defined using partial ordering on the dual cone $\mathbf{K}_*$. The dual cone comprises of vectors whose inner product with all vectors of the cone $\mathbf{K}$ are non-negative. The definition can be formally written as:

Definition 7.3 *Let $\boldsymbol{K} \in \mathcal{K}$ be a nonempty set. Then the dual cone $\boldsymbol{K}_*$ is given as,*

$$\boldsymbol{K}_* = \{y : y^T z \geq 0,\ \forall z \in \boldsymbol{K}\}.$$

The dual cone has some interesting properties -

Remark 7.4 *If $\boldsymbol{K}$ is a closed convex cone, then so is $\boldsymbol{K}_*$.*

Remark 7.5 *The dual cone to $\boldsymbol{K}_*$ is $\boldsymbol{K}$ itself, i.e. $(\boldsymbol{K}_*)_* = \boldsymbol{K}$.*

Examples of dual cones for some family of cones -

- $(\mathbb{R}^m_+)_* = \mathbb{R}^m_+$
- $(S^m_+)_* = S^m_+$
- $(\mathcal{L}^m)_* = \mathcal{L}^m$

Recall the strong and weak duality theorems statements discussed in previous lectures.

- Weak Duality - $Opt(P) \geq Opt(D)$
- Strong Duality - If P is strictly feasible and bounded then (D) is solvable and $Opt(D) = Opt(P)$.

We are interested in obtaining similar results for CP and its dual problem. A weak duality statement can be obtained easily.

Theorem 7.6 (Weak Duality) *The optimal value of (D) is a lower bound on the optimal value of (P).*

Proof: If x is a feasible solution to CP then we have $A^T x \succcurlyeq_{\mathbf{K}} b$ (or $Ax - b \in \mathbf{K}$). And for an admissible vector y belonging to the dual cone ($y \in \mathbf{K}_*$) we get the scalar equation, $y^T(Ax - b) \geq 0 \implies y^T Ax \geq y^T b$.

If y is a feasible solution to the dual problem (D), then we have $A^T y = c$.

Clearly, using the conditions obtained above,

$$c^T x = (A^T y)^T x = y^T Ax \geq y^T b = b^T y$$

$$c^T x \geq b^T y$$

The condition derived above is valid for all feasible solutions of (P) and (D). Hence, the minimum value of $c^T x$ or the solution to the primal CP or (P) is lower bounded by the maximum value of $b^T y$ or the solution to the dual problem (D). [1] ■

A stronger argument can indeed be made for a conic program (P) and its dual (D) (see Conic Duality Theorem, (Theorem 2.4.1) in [BTN01]). The optimality conditions for the primal-dual solution are characterized by the following statement.

[1] A more inquisitive reader is directed to Section 2.3-2.4 of [BTN01] for a more thorough treatment of duality.

Theorem 7.7 (Optimality Conditions) *If one of (P) or (D) is strictly feasible then (x_*, y_*) is optimal solution to the Primal-Dual problem,*

- *if and only if $c^T x_* - b^T y_* = 0$ [Zero Duality Gap]*
- *if and only if $y_*^T (Ax_* - b) = 0$ [Complimentary Slackness]*

Proof: We begin by first proving that (x_*, y_*) is optimal solution is equivalent to the zero duality gap.

$$\begin{aligned} c^T x_* - b^T y_* &= c^T x_* - b^T y_* - Opt(P) + Opt(P) + Opt(D) - Opt(D) \\ &= \underbrace{\left[c^T x_* - Opt(P)\right]}_{\geq 0} + \underbrace{\left[Opt(D) - b^T y_*\right]}_{\geq 0} + \underbrace{Opt(P) - Opt(D)}_{\geq 0} \quad \ldots \text{Weak Duality, Theorem 7.6} \end{aligned}$$

The duality gap is always greater than or equal to zero, and it is exactly zero if and only if $Opt(P) = Opt(D)$. Clearly the duality gap being zero is necessary and sufficient for the primal-dual variables to be optimum.

We can obtain complimentary slackness condition directly from duality gap condition. We assume that the solution (x_*, y_*) is feasible and we get,

$$\begin{aligned} c^T x_* - b^T y_* &= (A^T y)^T x_* - b^T y_* \\ &= y^T \left[Ax_* - b\right] = 0 \qquad \ldots \text{if and only if } c^T x_* - b^T y_* = 0 \end{aligned}$$

Hence the complimentary slackness being zero is also necessary and sufficient for the primal-dual variable to be the optimum. ■

7.2 Interior Point Methods

Interior Point Methods are a class of algorithms that solve linear and nonlinear convex optimization problems [BV04,DT06,NT08]. Interior point method was first introduced by John von Neumann (in discussion with George Dantzig) in 1948 [DT06], however the method was inefficient and slower in practice as compared to the Simplex method. Karmarkar proposed a provable polynomial-time algorithm (called Karmarkar's algorithm) for linear programming problems in 1984 [K84]. It was much more efficient than Dantzig's simplex method and allowed solving a larger class of optimization problems. Karmarkar's improvements revitalized the study of interior point algorithms. And with further refinements, the Interior Point Methods have become extremely efficient and have found applications in virtually every field of continuous optimization.

7.2.1 Barrier Method

Interior point methods typically solve the constrained convex optimization problem by applying Newton's method to a sequence of equality constrained problems. We will study a particular interior point algorithm, the barrier method, in this lecture. Consider an optimization problem as shown below:

$$\begin{aligned} \min \quad & c^T x \\ s.t. \quad & x \in X. \end{aligned} \tag{7.4}$$

Barrier methods as the name suggest employ barrier functions to integrate inequality constraints into the objective function. First, we motivate the definition and development of barrier functions.

Barrier function. Since we want to merge inequality constraints to the objective, the following conditions on barrier functions seem natural

- Barrier function should map the feasible set $\text{int}(X)$ in (7.4) to real space, i.e. $F : \text{int}(X) \to \mathbb{R}$.
- $F(x)$ is integrated into the objective function and we intend to use gradient based method, so the barrier function should be reasonably "smooth" (continuously differentiable).
- $F(x)$ should behave like a barrier and grow unbounded as the state (x) enters the boundary of the feasible set, i.e. $F(x) \to +\infty$ if $x \in \partial X$.
- The Hessian (or second derivative for functions of scalars) needs to be positive definite, $\nabla^2 F(x) \succ 0$, $\forall x \in X$.

One would, almost unconsciously, gravitate towards indicator functions as potential barrier functions to integrate the constraint in Problem (7.4) with the objective function (see Section 11.2 in [BV04]). However, the non-smooth and non-differentiable nature of indicator functions makes them unsuitable for use as barrier functions. Logarithmic functions satisfy the conditions stated above. Furthermore, their ability to grow exponentially almost mimics the jump seen in indicator functions.

Examples of barrier functions

- $X = \mathbb{R}^m$, $F(x) = -\sum_{i=1}^{m} \log(x_i)$
- $X = \mathcal{L}^m$, $F(x) = -\log(x_m^2 - \sum_{i=1}^{m-1} x_i^2)$
- $X = S_+^m$, $F(x) = -\log(\det(X))$
- For p inequality constraints of the type $f_i(x) \leq 0$, $F(x) = -\sum_{i=1}^{p} \log(f_i(x))$ can be used as the Barrier function.

The basic idea behind Barrier methods is as follows, we first rewrite the constrained convex optimization problem in 7.4 as follows,

$$\min \quad c^T x + \frac{1}{t} F(x) \tag{7.5}$$

where t is a parameter. If $t < \infty$ then the solution $x(t)$ to the above Problem (7.5) belongs to the interior of X, i.e. $x(t) \in \text{int}(X)$, and as t increases and $t \to \infty$, the solution to the above problem tends to the optimum, i.e. $x(t) \to x_*$. The Barrier method can be expressed in a pseudo-code see Algorithm 1.

Path Following Scheme

We can use the following equation to update t.

$$t_{k+1} = (1 + \frac{1}{13\sqrt{\nu}}) t_k \tag{7.6}$$

And use Newton's method in a sequential fashion to compute the minimum to Problem (7.5).

$$x_{k+1} = x_k - \left[\nabla^2 F(x_k)\right] \left[t_k c + \nabla F(x_k)\right] \tag{7.7}$$

Note that ν is a parameter associated with the Barrier function.

Algorithm 1 Barrier Method (Adapted from [BV04])

Input: Objective function $f(x)$, Barrier function $F(x)$, $x \in \text{int}(X)$,
    t Update Equation, Stopping Criteria (dependent on tolerance (ϵ))
Result: ϵ-suboptimal solution to Problem 7.4
repeat
    Centering Step: Compute $x_*(t) = \text{argmin}\{f(x) + (1/t)F(x)\}$ ▷ Use Eq. (7.7)
    Update: $x = x_*(t)$
    Increase t: Use t Update Equation ▷ Use Eq. (7.6)
until Stopping Criteria

Convergence

The error $\epsilon(x_k)$ is upper bounded by the following relation,

$$\epsilon(x_k) \triangleq \|c^T x_k - \min_{x \in X} c^T x\| \leq \frac{\left(2\nu e^{-\frac{k}{1+13\sqrt{\nu}}}\right)}{t_0}. \tag{7.8}$$

The number of steps or iterations required for the error function to drop below ϵ_0 is given by,

$$k \sim \mathcal{O}(\nu \log(\frac{\nu}{\epsilon_0})) \quad \text{for } \epsilon(x_k) \leq \epsilon_0 \tag{7.9}$$

The computational complexity of the barrier method is $\mathcal{O}(M\nu \log(1/\epsilon_0))$, where M is the cost of Newton step, usually of order $O(n^3)$. However the value of n can be quite large for large data sets. Algorithms dependent on n and higher powers of n significantly slow down with large datasets. Operations such as evaluating the Hessian is particularly expensive when n is large. It is hence critical to design algorithms with *dimension-free* complexity , if we wish to perform learning/inference/analytics over large data sets.

7.3 Introduction to Optimization Algorithms

A general optimization problem is defined as,

$$\begin{aligned} \min \quad & f(x) && \\ s.t. \quad & x \in X && \\ & g_i(x) \leq 0 && i = 1, \ldots, m \\ & h_j(x) = 0 && j = 1, \ldots, p \end{aligned} \tag{7.10}$$

where, $f(x)$ is the objective function, X is the feasible set, $g_i(x) \leq 0$ represents m inequality constraints and $h_j(x)$ represents p equality constraints. Recall from Definition 5.1, that convex optimization problems have linear equality constraints (or do not have equality constraints at all). In this section, we will look at a general taxonomy for optimization algorithms and few remarks on measures of error.

7.3.1 Black Box Oriented Algorithms

Black box oriented algorithms are algorithms that generate sequence of iterates $x_1, x_2, \ldots$, such that any new iterate x_{t+1} utilizes only *local information* collected along the search path (information at x_k or before). Unlike Interior Point Method, black box oriented algorithms usually do not exploit the structure of the problem.

Often analytic forms of the objective and/or constraint functions are unavailable and we may have to resort to using just function and gradient values at specified states [BBA]. Many optimization problems in finance (portfolio optimization), bioinformatics (protein folding), system theory (data drive system identification) etc. belong to this type. Black box algorithms are naturally suited for such problems. The optimizer queries the system to get function and gradient values at specific points (iterates) and then uses just this information (local) to perform optimization.

Depending on the information utilized, optimization algorithms can be classified into zero-order, first-order or second-order methods.

- **Zero-Order Methods** Optimization algorithms that utilize only function ($f(x_t)$) and constraint ($g_i(x_t)$) values are categorized as Zero-Order methods. Algorithms such as Nelder-Mead method, Random Search, Golden Section Search are Zero-Order methods. Several population based heuristic methods such as Particle Swarm Optimization (PSO), Genetic Algorithms (GA), Simulated Annealing (SA) are also examples of Zero-Order methods. In population based methods, we start with a population of feasible candidate solutions, and then prune and improve the population using function values.
- **First-Order Methods** Optimization algorithms that utilize subgradients (or gradient) of the function ($\partial f(x_t)$) and constraint ($\partial g_i(x_t)$) along with function ($f(x_t)$) and constraint ($g_i(x_t)$) values are called First-Order Methods. Gradient Descent, Stochastic Gradient Descent, Coordinate Descent, Frank-Wolfe are some of the popular First-Order algorithms. First-order methods are extremely popular in large scale optimization, machine learning etc. due to low computational complexity and speed.
- **Second-Order Methods** Optimization algorithms that utilize hessian information ($\nabla^2 f(x_t)$, $\nabla^2 g_i(x_t)$), gradients information ($\nabla f(x_t)$, $\nabla g_i(x_t)$), and function and constraint ($f(x_t)$) ($g_i(x_t)$) values are called Second-Order Methods. Newton Method, BFGS method etc are some popular examples of the second-order methods.

Later in the course we will analyze algorithms to obtain theoretical guarantees on the correctness, performance and convergence speed of the optimization algorithms. Since large scale optimization problems (machine learning etc.) involve huge datasets with millions of model parameters to optimize for, it becomes extremely critical for the optimization algorithm to be computationally "simple". We will also analyze the trade-offs between convergence speed, computational complexity and optimality for optimization algorithms.

7.3.2 Measure of Error

A measure of error needs to be defined to evaluate the quality of solution obtained from any optimization algorithm (especially iterative algorithms). Let us denote error at solution x_t by $\epsilon(x_t)$. Note that we would want error to be non-negative ($\epsilon(x_t) \geq 0$) and it should approach zero as the solution approaches optima ($x_t \to x_* \implies \epsilon(x_t) \to 0$). We will list down a few metrics that satisfy the above conditions:

1. $\epsilon(x_t) = \inf_{x_* \in X_*} \|x_t - x_*\|$
   This metric represents the distance between the solution (x_t) and the optimal solution (x_*). It clearly satisfies both criteria mentioned above.
2. $\epsilon(x_t) = f(x_t) - f(x_*)$
   Here, $\epsilon(x_t)$ represents the mismatch between function values at a feasible solution ($f(x_t)$) and the optimum ($f(x_*)$). Clearly this metric possesses both the properties mentioned above. As $x_t \to x_*$, the function value $f(x_t) \to f(x_*)$ and $\epsilon(x_t) \to 0$.

3. $\epsilon(x_t) = \max(f(x_t) - f(x_*), [g_1(x_t)]_+, [g_2(x_t)]_+, \ldots, [g_m(x_t)]_+)$
   Note $[g_i(x_t)]_+ = \max(g_i(x_t), 0)$. In this metric, we try to bound both the optimality and infeasibility. This ensures that if $\epsilon(x_t) < \epsilon_0$ then both the function distance from optimum is less than ϵ_0 and the maximum constraint slackness is less than ϵ_0. That is, $\epsilon(x_t) < \epsilon_0 \implies f(x_t) < f(x_*) + \epsilon_0$ and $g_i(x_t) < \epsilon_0 \ \forall i = 1, \ldots, m$.

References

[NT08] A. Nemirovski and M. Todd, "Interior-point methods for optimization," *Acta Numerica*, Vol 17, 2008, pp. 191–234.

[K84] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, Vol 4, 1984, pp. 373-395.

[BV04] S. Boyd and L. Vandenberghe, "Convex Optimization," Cambridge University Press, 2004.

[BTN01] A. Ben-Tal and A. Nemirovski, "Lectures on Modern Convex Optimization: Analysis, Algorithms and Engineering Applications," MPS-SIAM Series on Optimization,SIAM, 2001.

[DT06] G. B. Dantzig and M. N. Thapa, "Linear programming 2: theory and extensions," Springer Science & Business Media, 2006.

[DT06] D. G. Luenberger and Y. Ye, "Linear and Nonlinear Programming," Springer Science & Business Media, 2008.

[BBA] Black Box Algorithms, *http://archimedes.cheme.cmu.edu/?q=bbo* [Accessed: 09/13/16].