EE 381V: Large Scale Optimization **Fall 2012**

Lecture 16 — October 23

Lecturer: Caramanis & Sanghavi *Scribe: Chris Slaughter, Ioannis Mitliagkas*

Overview:

1. solving the primal via the dual
2. sensitivity analysis
3. semidefinite programming

16.1 Solving the Primal from the Dual

We analyze a simple problem where we can solve the dual much more easily than the primal. We also recover the primal variables from the solution. Consider the following problem:

$$\begin{aligned} \min \quad & \textstyle\sum_i f_i(x_i) && (16.1)\\ s.t. \quad & a^T x = b && (16.2) \end{aligned}$$

We can compute the Lagrangian as

$$\begin{aligned} \Lambda(x;\ v) &= \sum_i f_i(x_i) + v(a^T x - b) && (16.3)\\ &= \sum_i (v a_i x_i + f_i(x_i)) - vb && (16.4) \end{aligned}$$

Computing the dual function,

$$\begin{aligned} g(v) &= \inf_x \Lambda(x;\ v) && (16.5)\\ &= -vb + \sum_i \inf_x (v a_i x_i + f_i(x_i)) && (16.6)\\ &= -vb - \sum_i f_i^*(-v a_i) && (16.7) \end{aligned}$$

Note that $f_i^*(\theta) = \sup_x(\theta^T x - f_i(x))$ is called the *convex conjugate* or *Fenchel conjugate* of f_i. In many cases, $f_i^*(\theta)$ is known or efficiently computable.

In conclusion we can solve the simple unconstrained dual optimization problem in one variable:

$$\max_{v} \quad -bv - \textstyle\sum_i f_i^*(-va_i) \tag{16.8}$$

By inspecting the first-order optimality conditions, we can recover the primal variables as $x_i^* = -va_i$.

This example illustrates a case where the dual problem is substantially more tractable than the primal problem. It is important to note that discovering a tractable dual problem not only depends on the primal problem, but also its optimization formulation. Hence some formulations may lead to more tractable duals.

16.2 Sensitivity Analysis

Given an optimization problem, sensitivity analysis asks how the optimal value changes if we perturb the constraints. Consider the MAX-CUT problem. MAX-CUT computes the maximum flow, or capacity, through a network of pipes. A practical design question is *which link capacities should be increased to increase maximum flow*. It turns out that the good candidate links are exactly those with active constraints. In this section, we make this idea more formal.

Consider an optimization problem in modified standard form:

$$\begin{aligned} p = \quad \min_x \quad & f_0(x) && (16.9)\\ s.t. \quad & f_i(x) \le u_i, \quad \forall\, i && (16.10)\\ & h_j(x) = v_j, \quad \forall\, j && (16.11) \end{aligned}$$

It is immediately clear that for $u_i \ge 0$, an increase in u_i leads to lower (better) p^*. We now prove the following lemma:

Lemma 16.1. *1 Let $p^*(u,v)$ be the optimal value of the above optimization problem with constraint parameters u and v, and let λ^*, μ^* be optimal dual variables of $p^*(0,0)$. Then:*

$$p^*(u,v) \ge p^*(0,0) - (\lambda^*)^T u - (\mu^*)^T v$$

Proof:

$$\begin{aligned} p^*(0,0) &= \Lambda(x^*, \lambda^*, \mu^*) && (16.12)\\ &\le \Lambda(x, \lambda^*, \mu^*) && (16.13)\\ &= f_0(x) + \sum_i \lambda_i^* f_i(x) + \sum_j \mu_j^* h_j(x) && (16.14)\\ &\le f_0(x) + \sum_i \lambda_i^* \mu_i + \sum_j \mu_j^* v_j && (16.15) \end{aligned}$$

□

The last line follows from the feasibility of x for (u, v). We now utilize the definition of derivative to analyze the sensitivity of a single constraint.

Lemma 16.2. *Given λ^* and μ^* as above,*

$$\frac{\partial p^*(u,v)}{\partial u_i} = \lambda_i^*$$

Proof:

$$p^*(u_i, 0) - p^*(0,0) \geq -\lambda_i^* u_i \tag{16.16}$$

Assuming $u_i > 0$,

$$\frac{p^*(u_i, 0) - p^*(0,0)}{u_i} \geq -\lambda_i \tag{16.17}$$

Assuming $u_i < 0$,

$$\frac{p^*(u_i, 0) - p^*(0,0)}{u_i} \leq -\lambda_i \tag{16.18}$$

By continuity of p^*, the positive and negative partial derivatives are equal at $(0, 0)$ and we obtain the lemma. □

16.2.1 Sensitivity of Max-Margin Linear Classifiers from KKT Conditions

Recall the max-margin linear classification problem:

$$\begin{aligned} \min \quad & ||w||_2^2 && (16.19)\\ s.t. \quad & y_i(w^T x - b) \geq 1 \ \ \forall\, i && (16.20) \end{aligned}$$

We know that the hyperplane normal can be recovered from dual variables as $w = \sum_i \alpha_i y_i x_i$. Yet not each data point x_i is required to define w. By complementary slackness, if $y_i(w^T x - b) > 1$, then necessarily $\alpha_i = 0$. But the converse does not hold. That is, if i is such that $y_i(w^T x - b) = 1$, it is not necessarily true $\alpha_i > 0$. It turns out that which α_i are non-zero is related to the dimensionality of the feature space and the number of data points on the margin. In an n-dimensional feature space, we have $n+1$ linear constraints on $\{\alpha_i\}$ per the equations for (w, b). Hence if more than $n+1$ data points lie on the margin, it is possible that for some i such that $y_i(w^T x - b) = 1$, the corresponding α_i is in fact 0.

16.3 Semi-Definite Programming

A very important sub-class of convex problems, that of *Semi-Definite Programming* problems can be formulated in its simplest form as:

$$\begin{aligned} \min \quad & c^T x && (16.21)\\ s.t. \quad & G + \sum_i x_i F_i \succeq 0, && (16.22) \end{aligned}$$

where G, $\{F_i\}_i$ are symmetric matrices. Constraints of this type are referred to as *Linear Matrix Inequalities*. SDPs have risen to popularity for being quite powerful and relatively easy to solve. Using modern solvers, dealing with SDPs is almost as simple as dealing with instances of Linear Programming.

Some problems can involve a number of constraints each of which can be formulated as an LMI. It is quite straight-forward to argue that the formulation involving a single LMI is as general. Consider the problem

$$\begin{aligned} \min \quad & c^T x && (16.23) \\ s.t. \quad & A + \sum_i x_i B_i \succeq 0 && (16.24) \\ & M + \sum_i x_i N_i \succeq 0. && (16.25) \end{aligned}$$

We can formulate it as an equivalent SDP with a single LMI using

$$G = \begin{bmatrix} A & 0 \\ 0 & M \end{bmatrix}, \quad F_i = \begin{bmatrix} B_i & 0 \\ 0 & N_i \end{bmatrix}. \tag{16.26}$$

The most general form of SDP optimizes over a variable X, which is itself a matrix. Then it receives the form:

$$\begin{aligned} \min \quad & \langle A, X \rangle && (16.27) \\ s.t. \quad & \langle X, F_i \rangle = b_i, \qquad \forall i && (16.28) \\ & X \succeq 0. && (16.29) \end{aligned}$$

16.3.1 The Dual

In this section we will calculate the dual of the standard SDP form, as described in (16.22). Using the Lagrangian we can equivalently write:

$$\inf_x \sup_{Z \succeq 0} c^T x - \langle Z, G + \sum_i x_i F_i \rangle \tag{16.30}$$

To verify the validity of the Lagrangian multipliers added here, recall that

$$X \succeq 0 \Leftrightarrow \langle A, X \rangle \geq 0, \qquad \forall A \succeq 0. \tag{16.31}$$

Finally, the dual can be written as:

$$\begin{aligned} \min \quad & -\langle G, Z \rangle && (16.32) \\ s.t. \quad & \langle F_i, Z \rangle = c_i, \qquad \forall i && (16.33) \\ & Z \succeq 0. && (16.34) \end{aligned}$$

16.3.2 Applications and Specializations

It is often useful to be able to reformulate weaker problem classes as SDP instances, especially when one would like to bestow those classic problems with an additional, applitcation specific, constraint in the form of an LMI. For example, it is very common for LPs or Quadratic Programs to be rewritten as SDPs, when there is need to optimimize under uncertainty for problem parameters (Robust Optimization).

Starting with Linear Programming,

$$\begin{aligned} \min \quad & c^T x && (16.35) \\ s.t. \quad & a^T x \leq b && (16.36) \end{aligned}$$

we can easily reformulate this as an SDP by picking appropriate diagonal matrices F and G.

Similarly, instances of Quadratically Constrained Quadratic Programming (QCPC) can be transformed into SDPs. Consider the general formulation of a QCQP:

$$\begin{aligned} \min \quad & f_0(x) && (16.37) \\ s.t. \quad & f_i(x) \leq 0, \quad \forall i, && (16.38) \end{aligned}$$

where, without loss of generality,

$$f_i(x) = (A_i x + b)^T (A_i x + b_i) - c_i^T x - d_i. \tag{16.39}$$

Recall that the Schur complement of a matrix

$$X = \begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix}, \tag{16.40}$$

is defined as $S = C - B^T A^{-1} B$. Also,

$$X \succeq 0 \Leftrightarrow A \succeq 0 \,\text{or}\, S \succeq 0 \tag{16.41}$$

and

$$\text{If}\, X \succeq 0 \,\text{then}\, A \succeq 0 \Leftrightarrow S \succeq 0. \tag{16.42}$$

Exploiting these properties, we can rewrite the same QCQP as an SDP:

$$\begin{aligned} \min_{x,t} \quad & t && (16.43) \\ s.t. \quad & \begin{bmatrix} I & A_0 x + b_0 \\ A_0 x + b_0 & c_0^T x + d_0 + t \end{bmatrix} \succeq 0 && (16.44) \\ & \begin{bmatrix} I & A_i x + b_i \\ A_i x + b_i & c_i^T x + d_i \end{bmatrix} \succeq 0. && (16.45) \end{aligned}$$