10-725/36-725: Convex Optimization **Spring 2015**

Lecture 17: Primal-dual interior-point methods part II

Lecturer: Javier Pena *Scribes: Pinchao Zhang, Wei Ma*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

17.1 Review from last lecture

Last time we discussed the primal-dual interior-point methods for linear programming. The primal problem is

$$\begin{aligned} \min \quad & c^T x \\ & Ax = b \\ & x \geq 0 \end{aligned} \tag{17.1}$$

And the dual problem is

$$\begin{aligned} \max \quad & b^T y \\ & A^T y + s = c \\ & s \geq 0 \end{aligned} \tag{17.2}$$

The primal linear programming problem is in standard form. The dual problem is also a linear programming. With the theorem of strong duality, if and only if the dual problem is feasible, the two optimal values match. Several things we defined

- $\mathcal{F}^0$:=$\{(x, y, s) : Ax = b, A^T y + s = c, x, s > 0\}$
- Given $x, s \in \mathbb{R}^n_+, \mu(x, s)$:=$\frac{x^T s}{n}$
- $\mathcal{N}_2(\theta) := \{(x, y, s) \in \mathcal{F}^0 : ||XS1 - \mu(x, s)1||_2 \leq \theta\mu(x, s)\}$

Intuitively, μ is some measurement of the duality gap. In other words. it measures how far we are form the optimal point.

Recall that the central path equations are

$$A^T y + s = c \tag{17.3a}$$
$$Ax = b \tag{17.3b}$$
$$XS1 = \tau 1 \tag{17.3c}$$
$$x, s \geq 0 \tag{17.3d}$$

The key step of the IPM algorithm is the Newton step

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \tau 1 - XS1 \end{bmatrix} \tag{17.4}$$

Ideally, the Newton step updates the points and finally finds the optimal solution for both primal and dual problems. The essence of the Newton step is linearizing the equation 17.3a,17.3b and 17.3c in the central path. One can think of Newton's method in two ways: an optimization method or an algorithm for solving equations. Here the Newton step is used in the second way.

The short-step path following algorithm requires that the initial point located in $\mathcal{F}^0$. To eliminate this requirement, an infeasible interior-point algorithm is proposed with a different Newton step

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = - \begin{bmatrix} r_c \\ r_b \\ XS1 - \tau 1 \end{bmatrix} \tag{17.5}$$

where $r_b = Ax - b$ and $r_c := A^T y + s - c$. Note that in both short-step path following algorithm and the infeasible interior-point algorithm, computing the Newton step is the most expensive step. Everything else is cheap in these two algorithms.

17.2 Semidefinite program

17.2.1 standard form and the dual

The standard form of semidefinite program is

$$\begin{aligned} \min_X \quad & C \bullet X \\ \text{subject to} \quad & \mathcal{A}(X) = b \\ & X \succeq 0 \end{aligned} \tag{17.6}$$

where $\mathcal{A} : \mathbb{S}^n \to \mathbb{R}^m$ is a linear map which is supposed to be surjective, $b \in \mathbb{R}^m$, $C \in \mathbb{S}^n$. Similar to linear programming, we can derive the dual problem of 17.6 as

$$\begin{aligned} \max_y \quad & b^T y \\ \text{subject to} \quad & \mathcal{A}^*(y) \preceq C \end{aligned} \tag{17.7}$$

Or equivalently, introduce a slack variable S

$$\begin{aligned} \max_{y,S} \quad & b^T y \\ \text{subject to} \quad & \mathcal{A}^*(y) + S = C \\ & S \succeq 0 \end{aligned} \tag{17.8}$$

Where $\mathcal{A}^* : \mathbb{R}^m \to \mathbb{S}^n$.

To better understand the semidefinite program, we need to pay more attention to $\mathcal{A}^*$ and $\mathcal{A}$. Typically, we

have

$$\mathcal{A}(X) = \begin{bmatrix} A_1 \bullet X \\ ... \\ A_m \bullet X \end{bmatrix} \tag{17.9}$$

$$\mathcal{A}^*(y) = \sum_i y_i A_i \tag{17.10}$$

17.2.2 Weak duality and strong duality

Like what we did with linear programming, we hope to check relationship between the primal and dual problems.

Theorem 17.2.1. *(Weak duality) Assume X is primal feasible and y is dual feasible. Then*

$$b^T y \leq C \bullet X$$

The proof of weak duality is almost the same as the weak duality for linear programming.

Theorem 17.2.2. *(Strong duality)Assume both primal and dual problems are strictly feasible. Then their optimal values are the same and they are attained.*

Recall that in linear programming, we almost always have strong duality. The only exception is that the primal problem is unbounded or infeasible. In seminidifinite programming, things are different. If we assume both primal and dual problems are strictly feasible, the strong duality holds.If we do not have this condition, then duality will break down. This is one of the distinction between linear programming and semidifinite programming.

Here are several examples where the strong duality does not hold

1. The first example is

$$\begin{aligned} \min \quad & 2x_{12} \\ & \begin{bmatrix} 0 & x_{12} \\ x_{12} & x_{22} \end{bmatrix} \succeq 0 \end{aligned} \tag{17.11}$$

   This problem is equivalent to a standard form semidifinite programming problem

$$\begin{aligned} \min \quad & C \bullet X \\ & A_1 \bullet X = b \\ & X \succeq 0 \end{aligned} \tag{17.12}$$

   where

$$C = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{17.13a}$$

$$A_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \tag{17.13b}$$

$$b = 0 \tag{17.13c}$$

The dual would be

$$\begin{aligned} \max \quad & 0 \cdot y \\ & \begin{bmatrix} y & 0 \\ 0 & 0 \end{bmatrix} \preceq \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \end{aligned} \tag{17.14}$$

To guarantee that $\begin{bmatrix} 0 & x_{12} \\ x_{12} & x_{22} \end{bmatrix} \succeq 0$, we must have $x_{12} = 0$. Hence the primal optimal value is 0.

Now consider the constraint in dual problem. To make $\begin{bmatrix} y & 0 \\ 0 & 0 \end{bmatrix} \preceq \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, we need $\begin{bmatrix} -y & 1 \\ 1 & 0 \end{bmatrix} \succeq 0$. This is impossible no matter what value y is. The dual problem is infeasible. So the strong duality does not hold in this example.

2. The second example is

$$\begin{aligned} \min \quad & x_{11} \\ & \begin{bmatrix} x_{11} & 1 \\ 1 & x_{22} \end{bmatrix} \succeq 0 \end{aligned} \tag{17.15}$$

Similarly, this is equivalent to a standard form semidifinite programming problem

$$\begin{aligned} \min \quad & C \bullet X \\ & A_1 \bullet X = b \\ & X \succeq 0 \end{aligned} \tag{17.16}$$

where

$$C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \tag{17.17a}$$

$$A_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{17.17b}$$

$$b = 2 \tag{17.17c}$$

The dual would be

$$\begin{aligned} \max \quad & 2y \\ & \begin{bmatrix} 0 & y \\ y & 0 \end{bmatrix} \preceq \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \end{aligned} \tag{17.18}$$

Dual optimal value is 0, attained at $y = 0$. Primal optimal value is 0, while it is not attained because we need to make x_{11} positive. Strong duality does not hold.

3. The third example is

$$\begin{aligned} \min \quad & a x_{22} \\ & \begin{bmatrix} 0 & x_{12} & 1 - x_{22} \\ x_{12} & x_{22} & x_{23} \\ 1 - x_{22} & x_{23} & x_{33} \end{bmatrix} \succeq 0 \end{aligned} \tag{17.19}$$

The dual problem of this example is

$$\begin{array}{ll} \max & 2y_2 \\ & \begin{bmatrix} y_1 & 0 & y_2 \\ 0 & 2y_2 & 0 \\ y_2 & 0 & 0 \end{bmatrix} \preceq \begin{bmatrix} 0 & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{array} \tag{17.20}$$

The primal optimal value is 1 which is attainable. The dual optimal value is 0 which is attainable as well.

From these three examples, we know that the strong duality for semidifinite programming requires more conditions.

17.2.3 Optimality conditions

Assume strong duality holds. Then the points X^* and (y^*, S^*) are respectively primal and dual optimal solutions if and only if (x^*, y^*, S^*) solves

$$\mathcal{A}(X) = b \tag{17.21a}$$
$$\mathcal{A}^*(y) + S = C \tag{17.21b}$$
$$XS = 0 \tag{17.21c}$$
$$X, S \succeq 0 \tag{17.21d}$$

These are like the KKT conditions for both the primal and dual problems. To maintain 17.21a, 17.21b and 17.21d, and aim for 17.21c, the interior-point methods for semidefinite programming are developed.

17.2.4 Interior-point methods for SDP

Let's start from the Barrier method for semidefinite programming. If we are going to do something like the Barrier method for linear programming, we approximate the primal SDP by

$$\begin{array}{ll} \max & C \cdot X - \tau \log(\det X) \\ \text{subject to} & \mathcal{A}(X) = b \end{array} \tag{17.22}$$

Here the barrier function $F : S^n_{++} \to \Re$ is:

$$F(X) = -\log\det X = -\log \prod_{j=1}^{n} \lambda_j(X) \tag{17.23}$$

$$= -\sum_{j=1}^{n} \log(\lambda_j(X)) \tag{17.24}$$

Recall the barrier function in linear programming is $-\sum_{j=1}^{n} \log x_j$, so we actually extend the function form $\Re^n$ to S^n_{++}, even though the log determinant function is just an kind of approximation, it does come up in some statistical models, so it's a key objective for positive semidefinite matrix.

And as we did last time, the barrier dual problem of SDP is:

$$\begin{array}{ll} \max & b^T y + \tau \log(\det S) \\ \text{subject to} & \mathcal{A}^\star(y) + S = C \end{array} \tag{17.25}$$

So the SDP also has the nice duality connection, and very much as we did for linear programming, the optimality conditions for both primal and dual problems are these conditions.

$$\mathcal{A}(X) = b \tag{17.26}$$
$$\mathcal{A}^*(y) + S = C \tag{17.27}$$
$$XS = \tau I \tag{17.28}$$
$$X, S \succeq 0 \tag{17.29}$$

Recall that if we have function $f(t) = -\log t$, then the derivative if $f'(t) = -\frac{1}{t}$. So for $F(X) = -\log\det(X)$, we have $\nabla F(X) = -X^{-1}$, then if we write down the KKT conditions, we will get the above optimality conditions.

So following the same idea in linear programming we did, we are going to chase points that are near central path when τ goes to zero. And like what we did last time, we need to find some kind of proximity to the central path, and we want to define how we update the parameters. Then we denote

$$\mathcal{F}^0 := \{(X, y, S) : \mathcal{A}X = b, \mathcal{A}^\star y + S = C, X, S \succ 0\} \tag{17.30}$$

which is the primal and dual strictly feasible points. Then

$$\mu(X, S) := \frac{X \cdot X}{n} \tag{17.31}$$

is the gap between the pair of dual and primal problems. Then we use $d_F(X, S)$ as some kind of measure of proximity of XS to μI. So we don't require XS be equal to μI, instead we allow small difference. There are deep mathematics about how to measure the difference, basically we are comparing the eigenvalue of the matrix, here is the definition:

$$\begin{aligned} d_F(X, S) &:= \|\lambda(XS) - \mu(X, S)\|_2 && (17.32)\\ &= \left\|X^{1/2}SX^{1/2} - \mu(X, S)I\right\|_F && (17.33)\\ &= \left\|S^{1/2}XS^{1/2} - \mu(X, S)I\right\|_F && (17.34)\end{aligned}$$

After we have the idea of how to define the difference, we define the local neighborhood $\mathcal{N}_F(\theta)$ as

$$\mathcal{N}_F(\theta) := \{(X, y, S) \in \mathcal{F}^0 : d_F(X, S) \leq \theta_\mu(X, S)\} \tag{17.35}$$

Then if we have the Newton step to proceed the central path following procedures, we'll have we pretty similar interior-point algorithm for SDP. Here is the algorithm procedures:

1. Let $\theta, \delta \in (0, 1)$ be such that
$$\begin{aligned} \frac{7(\theta^2 + \delta^2)}{1 - \theta} &\leq \left(1 - \frac{\delta}{\sqrt{n}}\right)\theta && (17.36)\\ \frac{2\sqrt{2}\theta}{1 - \theta} &\leq 1 && (17.37)\end{aligned}$$
2. Let $(X^0, y^0, S^0) \in \mathcal{N}_F(\theta)$
3. For $k = 0, 1, \cdots$
Compute Nesterov-Todd direction for
$$\begin{aligned} (X, y, S) &= (X^k, y^k, S^k) && (17.38)\\ \tau &= \left(1 - \frac{\delta}{\sqrt{n}}\right)\mu(X, S) && (17.39)\end{aligned}$$

Set

$$(X^{k+1}, y^{k+1}, S^{k+1}) := (X^k, y^k, S^k) + (\Delta X, \Delta y, \Delta S) \tag{17.40}$$

And here is the convergence theorem, which is almost identical to what we did in linear programming.

Theorem 17.2.3. *The sequence generated by Algorithm SPF satisfies*

$$(X^k, y^k, S^k) \in \mathcal{N}_F(\theta)$$

and

$$\mu(X^{k+1}, S^{k+1}) = \left(1 - \frac{\delta}{\sqrt{n}}\right) \mu(X^k, S^k)$$

Corollary 17.2.3.1. *In* $\mathcal{O}\left(\sqrt{n}\log(\frac{n\mu(X^0,S^0)}{\epsilon})\right)$ *the algorithm yields* $(X^k, y^k, S^k) \in \mathcal{F}^0$ *such that*

$$C \cdot X_k - b^T y_k \leq \epsilon$$

We also have the similar results for long step and infeasible step methods.

17.2.5 Nesterov-Todd direction

To get the Newton step, we want to solve the system of equations. For the feasible method,

$$\mathcal{A}^\star(y) + S - C = 0 \tag{17.41}$$
$$\mathcal{A}(X) - b = 0 \tag{17.42}$$
$$XS = \tau I \tag{17.43}$$

The third formula is hard to satisfy, so what we do next is a kind of linearization. First we rewrite the equations above as:

$$G(X, y, S) - \begin{pmatrix} 0 \\ 0 \\ \tau I \end{pmatrix} = 0 \tag{17.44}$$

Then we linearize G, the first two equations are already linear, so we only need to care about XS, then we can write $S\Delta X + X\Delta S - \tau I - XS = 0$, which is the natural linearization, then we can write the Newton step as:

$$\begin{pmatrix} 0 & \mathcal{A}^\star & I \\ \mathcal{A} & 0 & 0 \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \Delta X \\ \Delta y \\ \Delta S \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \tau I - XS \end{pmatrix} \tag{17.45}$$

This linearization involves some symmetric issues, which may lead the update no longer in the feasible domains, so we need a more careful linearization. Here comes the Nesterov-Todd direction.

If we write the equation as

$$S - \tau X^{-1} = 0 \tag{17.46}$$

Then the linearization is

$$\tau X^{-1} \Delta X X^{-1} + \Delta S = \tau X^{-1} - S \tag{17.47}$$

where we use the fact that $\partial X^{-1} = X^{-1}\Delta X X^{-1}$. Also we have

$$X - \tau S^{-1} = 0 \tag{17.48}$$

So

$$\Delta X + \tau S^{-1}\Delta S S^{-1} = \tau S^{-1} - X \tag{17.49}$$

Then we claim that the proper primal-dual linearization will the average both equations. Define W as the geometric mean of X, S,

$$W = S^{-1/2}\left(S^{1/2}XS^{1/2}\right)S^{-1/2} \tag{17.50}$$

$$= X^{-1/2}\left(X^{1/2}SX^{1/2}\right)X^{-1/2} \tag{17.51}$$

So

$$W^{-1}\Delta X W^{-1} + \Delta S = \tau X^{-1} - S \tag{17.52}$$

or equivalently

$$\Delta X + W\Delta S W = \tau S^{-1} - X \tag{17.53}$$

Once we have this linearization, we can calculate the Newton step and proceed the interior-point methods.

17.3 Self-scaled cones

We can use the interior-point method even further to a class of cones called self-scaled cones. The idea is we can think of both linear and semidefinite programming as the special cases. For the linear programming, the cone is $\Re^n_+$ and for the semidefinite programming the cone is S^n_{++}. Then all we built for linear and semidefinite programming can be taken farther of the cones to the conic programming problems, as long as we define a good barrier function.

For the second order cones, the barrier function is like:

$$F(x) = -\log(x_0^2 - \|\bar{x}\|^2) \tag{17.54}$$

Then all the technical parts are similar to what we did for linear programming and semidefinite programming.

17.4 Conic Programming Solvers

Here we talk a little about how these theory are implemented. We can think about the matrix as a n^2 long vector.

$$X = [X_{11}\ X_{22}\cdots X_{nn}]^T \tag{17.55}$$

Which is called `vec` mapping in MATLAB, then `mat` is the inverse of `vec`. Another related mapping is `svec`:

$$X = \left[X_{11}\ \sqrt{2}X_{12}\cdots\sqrt{2}X_{1n}\ X_{22}\cdots\sqrt{2}X_{n-1,n}\ X_{nn}\right]^T \tag{17.56}$$

These we can write

$$X \cdot S = \texttt{vec}(X)^T\texttt{vec}(S) \tag{17.57}$$

$$= \texttt{svec}(X)^T\texttt{svec}(S) \tag{17.58}$$

Then here is an introduction to one of the existing solvers.

17.4.1 SeDuMi

SeDuMi can be used in MATLAB and has simple syntax. SeDuMi performs very well for small problems, but for large-scale problems, interior-point method meet a major bottleneck for computation. That is because of the hardness and solve the linear system to get the Newton step since the Hessian matrix is very complicated.