Convex Optimization
Review Session 4

Question 1 (Understanding Duality)

In this 'question' (it won't be much of a question – I won't ask much!) we'll try to get a better grip on the concept of duality.

Part A

In this part, we'll look at the Lagrangian as a relaxation of hard penalty functions. This way of looking at duality is particularly convenient when you actually have to write the dual of a particular program, because it gives you a quick way to determine the correct signs for the Lagrange multipliers.

Solution

[1]Consider, for the sake of argument, the optimization problem

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0} \\ & \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0} \end{aligned}$$

Where $\boldsymbol{g}(\boldsymbol{x})$ is a vector function – each entry of the vector denotes a different constraint.

An equivalent way of formulating this problem is[2]

$$\min \left[f(\boldsymbol{x}) + I_{-}(\boldsymbol{g}(\boldsymbol{x})) + I_0(\boldsymbol{h}(\boldsymbol{x})) \right] \tag{1}$$

We've simply eliminated the constraints and adding an infinite penalty whenever they're violated.

Unfortunately, this problem is rather difficult to solve. Let's try, therefore, and replace the 'hard' functions I_{-} and I_0 by 'softer' functions. In particular

- We can replace $I_{-}(\boldsymbol{u})$ with the function $\boldsymbol{\lambda} \cdot \boldsymbol{u}$, where $\boldsymbol{\lambda} \geq \boldsymbol{0}$. Like the 'real' displeasure function I_{-}, this penalizes a solution $\boldsymbol{x}$ if $\boldsymbol{g}(\boldsymbol{x})$ ends up being greater than $\boldsymbol{0}$ (because $\boldsymbol{\lambda}$ is positive).

  The problem is, of course, that this modified function also 'favors' our solution if $\boldsymbol{g}(\boldsymbol{x})$ ends up being *less* than zero,

[1]This treatment follows section 5.1.4 in Boyd

[2]I_0 is the indicator function for the set $\boldsymbol{0}$ – it is equal to 0 if its argument is $\boldsymbol{0}$ and ∞ otherwise. I_{-} is the indicator function for $\mathbb{R}_{-}$. It is equal to 0 if its argument is $\leq \boldsymbol{0}$, and ∞ otherwise.

which isn't quite right – in the original problem, we should be *indifferent* to deviations of $\boldsymbol{g}$ below 0.

- We can replace $I_0(\boldsymbol{u})$ with the function $\boldsymbol{\nu} \cdot \boldsymbol{y}$, where $\boldsymbol{\nu}$ takes *any* value. Clearly, this is once again a very poor approximation, because even though the function penalizes deviation of $\boldsymbol{h}(\boldsymbol{x})$ from $\mathbf{0}$ in one direction (depending on the sign of $\boldsymbol{\nu}$), it *favors* it in the other direction!

Nevertheless, let's make the change. The program in equation 1 then becomes precisely

$$L(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \min\left[f(\boldsymbol{x}) + \boldsymbol{\lambda} \cdot \boldsymbol{x} + \boldsymbol{\nu} \cdot \boldsymbol{x}\right]$$

With domain $\boldsymbol{\lambda} \geq \mathbf{0}$. This is none other than the Langrangian. So in some sense, the Lagrangian is obtained by relaxing the 'hard' penalty constraints I_- and I_0. The dual then consists of maximizing over *all* these possible parameters we could have used to approximate those functions.

Of course, the soft functions are rather poor estimators of I_0 and I_-. However, they do have one thing going for them – they'll always *underestimate* the true function[3]. This immediately gives weak duality. If it so happens that there's *just* that right combination of $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$ that hit a 'sweet spot', the result will end up being the same. Why convexity should lead to this 'sweet spot', though, is left to be determined!

[3]Go back and think about it if it's not obvious. Wherever the hard constraint is ∞, the soft constraint is finite, and wherever the hard constraint is 0, so is the soft constraint.

Part B

In this part, we'll try and understand exactly *what* we're doing when we find the Lagrangian and the dual of an optimization problem.

Solution

Consider the following (very simple!) optimization problem

$$\begin{aligned} \min \quad f(x) = \quad & (x+2)^2 + 4 \\ \text{subject to} \quad & x \geq 0 \end{aligned}$$

The objective function and feasible region are illustrated in the margin [TODO]. This is a very simple problem – from the diagram, it's pretty clear that the solution occurs at $x = 0$, with value 4.

Consider, however, the Lagrangian for this problem. It takes the form[4]

$$\mathcal{L}(x, \mu) = (x+2)^2 + 4 - \mu x$$

with $\mu \geq 0$.

[4]The more eagle-eyed among you will notice that in this case, $\mathcal{L}(x, \mu) = -f^*(\mu)$. I don't want to involve convex conjugates here, though, so as not to complicated matters unnecessarily.

Now consider finding the dual function. In this case, it's given by

$$\begin{aligned} q(\mu) &= \min_x \mathcal{L}(x,\mu) \\ &= \min_x \left[(x+2)^2 + 4 - \mu x\right] \\ &= \min_x \left[f(x) - \mu x\right] \end{aligned}$$

Looking at this expression carefully, we see that it's just trying to minimize the difference between the function $f(x)$ and the function μx. Put differently it finds the point at which the line $-\mu x$ (though the origin) and the function $f(x)$ are closest – in other words, it pushes that line upwards, and when it *just* touches the function (at the said closest point) it stops. This is illustrated in diagram [TODO].

What does the dual program do? It simply seeks to *maximize* this quantity over *all* possible *positive* gradients (ie: all possible hyperplanes) μ. Because the gradients are restricted to be *positive*, this will indeed find the optimum we require. This is illustrated in figure [TODO].

■ □ ■

Question 2 (Linear Discrimination and Duality)

Consider two sets of points

$$X = \{\boldsymbol{x}^1, \cdots, \boldsymbol{x}^N\} \subseteq \mathbb{R}^n \qquad Y = \{\boldsymbol{y}^1, \cdots, \boldsymbol{y}^M\} \subseteq \mathbb{R}^n$$

We wish to find a linear function $f(\boldsymbol{x}) = \boldsymbol{a} \cdot \boldsymbol{x} - b$ such that

$$f(\boldsymbol{x}^i) > 0 \quad \forall i \qquad f(\boldsymbol{y}^j) < 0 \quad \forall j \tag{2}$$

Such a function is called a *linear classifier*, because it is capable, given a new point, to "classify" it as either part of the X set or part of the Y set.

Because of the form of f and since the inequalities are strict, the set of inequalities in equation 2 are equivalent to the equations

$$\boldsymbol{a} \cdot \boldsymbol{x}^i - b \geq 1 \quad \forall i \qquad \boldsymbol{a} \cdot \boldsymbol{y}^j - b \leq 1 \quad \forall j \tag{3}$$

In this form, the inequalities imply that all the points in X lie on one side of the hyperplane $\boldsymbol{a} \cdot \boldsymbol{x} - b = 1$, and all the Y points lie on the other side of the hyperplane $\boldsymbol{a} \cdot \boldsymbol{x} - b = -1$. Of course, there are many such hyperplanes. Ideally,however, we'd like these two hyperplanes to be as *far* from each other as possible, to leave as much of a 'buffer region' as possible that separates the points.

Part A

Show that the following program finds the hyperplane $(\boldsymbol{a}, b)$ that separates X and Y 'as much as possible' (in the sense described above).

$$\begin{array}{ll} \min & -t \\ \text{s.t.} & \boldsymbol{a} \cdot \boldsymbol{x}^i - b \geq t \\ & \boldsymbol{a} \cdot \boldsymbol{y}^i - b \leq -t \\ & \|\boldsymbol{a}\|_2 \leq 1 \end{array}$$

Solution

First, consider the two hyperplanes $\boldsymbol{a} \cdot \boldsymbol{x} + b = 1$ and $\boldsymbol{a} \cdot \boldsymbol{x} - b = -1$. How do we find the *distance* between these two hyperplanes? Consider a point $\boldsymbol{x}_0$ on the -1 hyperplane. Clearly, to find the shortest (ie: perpendicular) distance to the $+1$ hyperplane, we need to move along the perpendicular vector $\boldsymbol{a}$ until we hit this other hyperplane. In other words, the distance is proportional to τ, where

$$\boldsymbol{a} \cdot (\boldsymbol{x}_0 + \tau \boldsymbol{a}) - b = 1$$

We know, however, that $\boldsymbol{a} \cdot \boldsymbol{x}_0 - b = -1$ (because $\boldsymbol{x}_0$ lies on the -1 plane), and so this reduces to

$$\tau = \frac{2}{\|\boldsymbol{a}\|_2}$$

As such, to find the desired hyperplane, we simply need to solve

$$\begin{array}{ll} \max & 2/\|\boldsymbol{a}\|_2 \\ \text{s.t.} & \boldsymbol{a} \cdot \boldsymbol{x}^i - b \geq 1 \\ & \boldsymbol{a} \cdot \boldsymbol{y}^i - b \leq -1 \end{array}$$

The objective isn't very nice! However, we can substitute

$$t = \frac{1}{\|\boldsymbol{a}\|_2} \qquad \tilde{\boldsymbol{a}} = \frac{\boldsymbol{a}}{\|\boldsymbol{a}\|_2} = t\boldsymbol{a} \qquad \tilde{b} = bt$$

to see that (after re-scaling the objective) the program above is equivalent to

$$\begin{aligned}\max \quad & t \\ \text{s.t.} \quad & \tilde{\boldsymbol{a}} \cdot \boldsymbol{x}^i - \tilde{b} \geq t \\ & \tilde{\boldsymbol{a}} \cdot \boldsymbol{y}^i - \tilde{b} \leq -t \\ & \|\tilde{\boldsymbol{a}}\|_2 = 1\end{aligned}$$

(the last constraint is needed because $\tilde{\boldsymbol{a}}$ is defined as a unit vector. Alternatively note that without that constraint the solution would blow up to infinity). This program is still not convex, because of the non-linear equality. However, a close look at the program should make it clear that it is in fact equivalent to

$$\begin{aligned}\max \quad & t \\ \text{s.t.} \quad & \tilde{\boldsymbol{a}} \cdot \boldsymbol{x}^i - \tilde{b} \geq t \\ & \tilde{\boldsymbol{a}} \cdot \boldsymbol{y}^i - \tilde{b} \leq -t \\ & \|\tilde{\boldsymbol{a}}\|_2 \leq 1\end{aligned}$$

This is because the way the program is structure tries to make $\boldsymbol{a}$ as large as possible – thus, the constraint will always be tight.[5]

[5] See Boyd exercise 8.23 for a more rigorous version of this argument.

Part B

Find the dual of the program in part A, and interpret it geometrically.

Solution

The Lagrangian of the above is[6].

[6] We let X be a matrix whose row contains the vectors in the set X, and similarly for Y

$$\begin{aligned}\mathcal{L}(\boldsymbol{a}, b, t, \boldsymbol{\lambda}, \boldsymbol{\mu}, \kappa) &= -t - \boldsymbol{\lambda} \cdot (X\boldsymbol{a} - b\mathbf{1} - t\mathbf{1}) \\ &\quad + \boldsymbol{\mu}(Y\boldsymbol{a} - b\mathbf{1} + t\mathbf{1}) + \kappa\left(\|\boldsymbol{a}\|_2 - 1\right)) \\ &= ([\boldsymbol{\lambda} + \boldsymbol{\mu}] \cdot \mathbf{1} - 1)t \\ &\quad + \left[Y^\top \boldsymbol{\mu} - X^\top \boldsymbol{\lambda}\right] \cdot \boldsymbol{a} + \kappa\|\boldsymbol{a}\|_2 \\ &\quad + (\boldsymbol{\lambda} \cdot \mathbf{1} - \boldsymbol{\mu} \cdot \mathbf{1})b \\ &\quad - \kappa\end{aligned}$$

Note the two key steps above – I first wrote the Lagrangian down, and then I re-write it to group all terms corresponding to each variable together. This, I have found, is the best strategy to successfully working out a dual, because it then allows you to more clearly see the implications of minimizing over each variable.

Minimizing over b implies that $(\boldsymbol{\lambda} \cdot \mathbf{1} - \boldsymbol{\mu} \cdot \mathbf{1}) = 0$, or else we can choose b very large and make the Lagrangian infinite. Similarly, minimizing over t implies that $[\boldsymbol{\lambda} + \boldsymbol{\mu}] \cdot \mathbf{1} = 1$, for similar reasons. As such, we find that

$$\boldsymbol{\lambda} \cdot \mathbf{1} = \boldsymbol{\mu} \cdot \mathbf{1} = \frac{1}{2}$$

When these hold, our Lagrangian becomes

$$\mathcal{L}(\boldsymbol{a}, b, t, \boldsymbol{\lambda}, \boldsymbol{\mu}, \kappa) = (Y^\top \boldsymbol{\mu} - X^\top \boldsymbol{\lambda}) \cdot \boldsymbol{a} + \kappa \|\boldsymbol{a}\|_2 - \kappa$$

We now need to minimize this with respect to $\boldsymbol{a}$. Clearly, if the norm 'dominates' the optimum will be at $\boldsymbol{a} = \mathbf{0}$. If the previous term 'dominates' then it might be possible to decrease the Lagrangian to $-\infty$. To make this more rigorous, consider that by the Cauchy-Schwartz Inequality, Now, consider that by the Cauchy-Schwarz inequality, we have that

$$|(Y^\top \boldsymbol{\mu} - X^\top \boldsymbol{\lambda}) \cdot \boldsymbol{a}| \leq \|Y^\top \boldsymbol{\mu} - X^\top \boldsymbol{\lambda}\|_2 \cdot \|\boldsymbol{a}\|_2$$

and that the inequality is tight. Thus, as long as

$$\|Y^\top \boldsymbol{\mu} - X^\top \boldsymbol{\lambda}\|_2 \leq \kappa$$

the Lagrangian will have a finite minimum, with a value of $-\kappa$.

Thus, our dual is

$$\begin{aligned} \max \quad & -\kappa \\ \text{s.t.} \quad & \|Y^\top \boldsymbol{\mu} - X^\top \boldsymbol{\lambda}\|_2 \leq \kappa \\ & \boldsymbol{\lambda}, \boldsymbol{\mu} \geq \mathbf{0} \\ & \boldsymbol{\lambda} \cdot \mathbf{1} = \boldsymbol{\mu} \cdot \mathbf{1} = \frac{1}{2} \end{aligned}$$

This can be written as

$$\begin{aligned} \max \quad & -\|Y^\top \boldsymbol{\mu} - X^\top \boldsymbol{\lambda}\|_2 \\ \text{s.t.} \quad & \boldsymbol{\lambda}, \boldsymbol{\mu} \geq \mathbf{0} \\ & \boldsymbol{\lambda} \cdot \mathbf{1} = \boldsymbol{\mu} \cdot \mathbf{1} = \frac{1}{2} \end{aligned}$$

Substituting $\tilde{\boldsymbol{\lambda}} = 2\boldsymbol{\lambda}$ and $\tilde{\boldsymbol{\mu}} = 2\boldsymbol{\mu}$, this becomes

$$\begin{aligned} \min \quad & \frac{1}{2}\|Y^\top \boldsymbol{\mu} - X^\top \boldsymbol{\lambda}\|_2 \\ \text{s.t.} \quad & \tilde{\boldsymbol{\lambda}}, \tilde{\boldsymbol{\mu}} \geq \mathbf{0} \\ & \tilde{\boldsymbol{\lambda}} \cdot \mathbf{1} = \tilde{\boldsymbol{\mu}} \cdot \mathbf{1} = 1 \end{aligned}$$

In this form, it becomes obvious that $X^\top \tilde{\boldsymbol{\lambda}}$ is a convex combination of the points in X – in other words, a point in the convex hull of

X. Similarly for $\tilde{\boldsymbol{\mu}}$ and Y. Thus, the dual program simply finds the two points in the convex hulls of X and Y that are as *close* to each other as possible. It of course makes sense that this should be equal to the optimal value of the primal – half the distance between the hyperplanes that strictly separates X and Y.

■ □ ■

Question 3 (Fitting a Quadratic with Constraints)

[1]Consider a set of inputs $\{\boldsymbol{x}^1, \cdots, \boldsymbol{x}^N\} \subseteq \mathbb{R}^n$, and a set of corresponding outputs $\{y_1, \cdots, y_n\} \subseteq \mathbb{R}$. We wish to fit a quadratic function

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top P\boldsymbol{x} + \boldsymbol{q} \cdot \boldsymbol{x} + r$$

to these data, so as to minimize the squared error of our fit.

Consider further that we impose the following constraints on f

- f must be concave.
- f must be non-negative over the box $\mathscr{B}$ given by

$$\mathscr{B} = \{\boldsymbol{x} : \boldsymbol{\ell} \leq \boldsymbol{x} \leq \boldsymbol{u}\}$$

- f must be non-decreasing on the same box $\mathscr{B}$.

Formulate this fitting problem as a convex optimization problem.

Solution

Our objective function will clearly be to minimize

$$\sum_{i=1}^{N} \left(f(\boldsymbol{x}^i) - y_1\right)^2$$

Consider that each of the terms in this objective are linear in the decision variables (P, $\boldsymbol{q}$ and r). As such, the entire objective is convex.

Let us now consider each of the constraints.

- $\boxed{f \text{ concave}}$. This is satisfied if and only if P is negative semidefinite. To see why, consider that the double derivative of f with respect to $\boldsymbol{x}$ is simply P. Thus,

$$P \preceq 0 \Leftrightarrow \nabla^2 f \preceq 0 \Leftrightarrow f \text{ concave}$$

  This constraint is a convex LMI, and so we're good.

[1]From a final exam given as part of Ciamac Moallemi's 'Foundations of Optimization' class, Fall 2010.

- $\boxed{f \text{ non-negative over } \mathscr{B}}$. Note that because f is non-decreasing over $\mathscr{B}$, the smallest value of f will occur at $\boldsymbol{x} = \boldsymbol{\ell}$. Therefore, this constraint can be written

$$f(\boldsymbol{\ell}) = \frac{1}{2}\boldsymbol{\ell}^\top P\boldsymbol{\ell} + \boldsymbol{q} \cdot \boldsymbol{\ell} + r \geq 0$$

This is linear in the decision variables P, $\boldsymbol{q}$ and r, and so it keeps out program convex.

- $\boxed{f \text{ non-decreasing over } \mathscr{B}}$. Consider that the derivative of f is given by

$$\nabla f(\boldsymbol{x}) = P\boldsymbol{x} + \boldsymbol{q}$$

We require this to be non-negative over $\mathscr{B}$. In other words, we require

$$\min_{\boldsymbol{x} \in \mathscr{B}} (P\boldsymbol{x} + \boldsymbol{q}) \geq 0 \tag{4}$$

One way to ensure this is true is to make sure that the condition holds for all $\boldsymbol{x}$ that is a corner of the box $\mathscr{B}$. However, the resulting program would contain a number of constraints exponential in the dimension n of the problem, since a box $\mathscr{B}$ in $\mathbb{R}^n$ has 2^n corners. Instead, consider the program on the LHS of constraint more carefully. Written in full, the LHS of its i^{th} component looks like (we let $\boldsymbol{p}^i$ be the i^{th} row of P)

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{p}^i \cdot \boldsymbol{x} + q_i \\ \text{s.t.} \quad & \boldsymbol{x} \leq \boldsymbol{u} \\ & \boldsymbol{x} \geq \boldsymbol{\ell} \end{aligned}$$

This is a standard LP, with dual

$$\begin{aligned} \max_{\boldsymbol{\lambda}^i, \boldsymbol{\mu}^i} \quad & \boldsymbol{u} \cdot \boldsymbol{\lambda}^i + \boldsymbol{\ell} \cdot \boldsymbol{\mu}^i + q_i \\ \text{s.t.} \quad & \boldsymbol{\lambda}^i + \boldsymbol{\mu}^i = \boldsymbol{p}^i \\ & \boldsymbol{\lambda}^i \leq \boldsymbol{0} \\ & \boldsymbol{\mu}^i \geq \boldsymbol{0} \end{aligned}$$

Now, consider that by strong duality, the objective value of this dual is always less than or equal to the value of the primal. Thus, the value of the primal is ≥ 0 if and only if there is a feasible solution to the dual that is ≥ 0. As such, constraint 4 can be written as n sets of constraints, each of which has the form

$$\begin{aligned} \boldsymbol{u} \cdot \boldsymbol{\lambda}^i + \boldsymbol{\ell} \cdot \boldsymbol{\mu}^i + q_i &\geq 0 \\ \boldsymbol{\lambda}^i + \boldsymbol{\mu}^i &= \boldsymbol{p}^i \\ \boldsymbol{\lambda}^i \leq \boldsymbol{0} \boldsymbol{\mu}^i &\geq \boldsymbol{0} \end{aligned}$$

These are linear constraints, and there are polynomially many of them, so we're golden!

■ □ ■

Question 4 (Industrial Spying)

[2]The PhD doesn't work out, and you decide to start a career as an industrial spy instead.

Your employer's main competitor produces m widgets using n machines. Let $\boldsymbol{x} \in \mathbb{R}^n$ denote the amount of time the competitor uses each machine. The matrix $A \in \mathbb{R}^{m \times n}$ describes the output of each machine in that a machine allocation $\boldsymbol{x}$ will produce $A\boldsymbol{x} \in \mathbb{R}^m$ widgets. The vector $\boldsymbol{c} \in \mathbb{R}^n$ denotes the running cost per unit time of each machine. Finally, the vector $\boldsymbol{b}^i \in \mathbb{R}^m$ denotes the demand for each widget in year i.

Clearly, the competitor chooses $\boldsymbol{x}^i$ (the machine allocation in year i) by solving the following LP

$$\begin{array}{ll} \min & \boldsymbol{c} \cdot \boldsymbol{x}^i \\ \text{s.t.} & A\boldsymbol{x}^i \geq \boldsymbol{b}^i \end{array}$$

Some subset $I \subset \{1, \cdots, n\}$ of the machines is standard and used across the widget industry – your employer therefore knows the corresponding costs $\{c_i\}_{i \in I}$. The rest of the machines, however, are proprietary to your competitor, and your boss is interested in knowing exactly how efficient his competitors machines are (in other words, he wants to know $\{c_i\}_{i \notin I}$). This secret is closely guarded, and your boss' spies have never been able to obtain these numbers. (Though your boss does know that $\boldsymbol{c}$ is constant over the years, as is A, since the design of the machines does not change.)

Thankfully, your has been able to gather *some* less sensitive intelligence over the last r years. For each of these years i, your boss knows the demand his competitor faced, $\boldsymbol{b}^i$, as well as the optimal machine allocation $\boldsymbol{x}^i$ it used in each year.

Given these data, find the tightest upper and lower bounds for each unknown component of c.

Solution

First, consider that it is essential for I not to be empty. If it is, then we can only determine $\boldsymbol{c}$ up to an arbitrary constant, since any solution $\boldsymbol{c}$ will also be optimal for $\alpha\boldsymbol{c}$, for any $\alpha > 0$.

That said, let's proceed. First, note that the dual of the LP in

[2]Adapted from Boyd, additional exercise 4.13

question is

$$\begin{aligned} \max \quad & \boldsymbol{b}^i \cdot \boldsymbol{\lambda}^i \\ \text{s.t.} \quad & \boldsymbol{c} = A^\top \boldsymbol{\lambda}^i \\ & \boldsymbol{\lambda}^i \geq \mathbf{0} \end{aligned}$$

This means that for any primal-dual solution pair $(\boldsymbol{x}^i, \boldsymbol{\lambda}^i)$, we must have

- $\boldsymbol{\lambda}^i$ is dual feasible:

$$\boldsymbol{c} = A^\top \boldsymbol{\lambda}^i \qquad \text{for some } \boldsymbol{\lambda} \geq \mathbf{0}$$

- The optimal solutions are equal

$$\boldsymbol{b}^i \cdot \boldsymbol{\lambda}^i = \boldsymbol{c} \cdot \boldsymbol{x}^i$$

Feeding our expression for $\boldsymbol{c}$ from the previous part into this and re-arranging, we obtain

$$\boldsymbol{\lambda}^i \cdot (\boldsymbol{b}^i - A\boldsymbol{x}^i) = 0$$

And since $\boldsymbol{\lambda} \geq \mathbf{0}$, every part of this sum must be equal to 0, so[7]

$$\lambda^i_j (b^i_j - \boldsymbol{a}^i_j \cdot \boldsymbol{x}^i) = 0$$

[7]Note that this is simply complementary slackness...

As such, to summarize, $\boldsymbol{c}$ is consistent with the observations if and only if the following constraints are satisfied

$$\begin{array}{rlll} \boldsymbol{\lambda}^i & \geq \mathbf{0} & i = 1, \cdots, r & \\ \boldsymbol{c} & = A^\top \boldsymbol{\lambda}^i & i = 1, \cdots, r & \\ \lambda^i_j (b^i_j - \boldsymbol{a}^i_j \cdot \boldsymbol{x}^i) & = 0 & i = 1, \cdots, r & j = 1, \cdots, n \\ c_i & = \text{know value} & i \in I & \end{array}$$

A careful look at these constraints reveals that they are in fact linear in the unknowns $\boldsymbol{\lambda}$ and $\boldsymbol{c}$. We can therefore maximize and minimize c_i for every $i \notin I$ subject to these constraints to find upper and lower bounds on the unknown components of $\boldsymbol{c}$.

■ □ ■

Question 5 (Homeland Security vs. Smugglers)

[3]A smuggler moves along a directed acyclic graph with m edges and n nodes, from a source node (which we take as node 1) to

[3]From the final to Garud Iyengar's 'Convex Optimization' course, Spring 2011

a destination node (which we take as node n), along some (directed) path. Each edge k has a detection failure probability p_k, which is the probability that the smuggler passes over that edge undetected. The detection events on the edges are independent, so the probability that the smuggler makes it to the destination node undetected is

$$\prod_{k\in\mathscr{P}} p_k$$

where $\mathscr{P} \subseteq 1, \cdots, m$ is (the set of edges on) the smugglers path. We assume that the smuggler knows the detection failure probabilities and will take a path that maximizes the probability of making it to the destination node undetected. We let $P_{\max}$ denote this maximum probability (over paths). (Note that this is a function of the edge detection failure probabilities.)

The edge detection failure probability on an edge depends on how much interdiction resources are allocated to the edge. Here we will use a very simple model, with $x_j \in \mathbb{R}_+$ denoting the effort (say, yearly budget) allocated to edge j, with associated detection failure probability

$$p_j = e^{-a_j x_j}$$

where $a_j \in \mathbb{R}_{++}$ are given. The constraints on the vector $\boldsymbol{x}$ are a maximum for each edge, $\boldsymbol{x} \leq \boldsymbol{x}^{\max}$, and a total budget constraint $\mathbf{1} \cdot \boldsymbol{x} \leq B$.

Homeland security's aim is to find the allocation of resources $\boldsymbol{x}$ that satisfies the constraints and minimize $P_{\max}$.

Show that this can be formulated as a convex optimization problem that is *not* exponential in the problem dimensions. How simple can you make the problem.

Solution

This is a pretty cool problem, and it's especially cool because it's actually no more than an LP!

To see how, first consider that homeland security's problem can be written as

$$\begin{aligned}
\min_{\boldsymbol{x}} \max_{\mathscr{P}} \quad & \prod_{k\in\mathscr{P}} e^{-a_j x_j} \\
\text{s.t.} \quad & \mathbf{0} \leq \boldsymbol{x} \leq \boldsymbol{x}^{\max} \\
& \mathbf{1} \cdot \boldsymbol{x} = 1 \\
& \mathscr{P} \text{ is a valid path from 1 to } n
\end{aligned}$$

Notice that this is a two-stage optimization problem – first, homeland security sets $\boldsymbol{x}$, and then, given that $\boldsymbol{x}$, the smuggler chooses the $\mathscr{P}$ that maximizes the escape probability.

Now, note that the objective can be written $\exp\left[-\sum_{k\in\mathscr{P}} a_j x_j\right]$. Furthermore, since exp is a monotonically increasing function, we can write our objective as

$$\min_{\boldsymbol{x}} \max_{\mathscr{P}} \left[-\sum_{k\in\mathscr{P}} a_j x_j\right]$$

Finally, taking the minus sign out of the problem, we find that this is equivalent to

$$\begin{aligned} \max_{\boldsymbol{x}} \min_{\mathscr{P}} \quad & \sum_{k\in\mathscr{P}} a_j x_j \\ \text{s.t.} \quad & \mathbf{0} \le \boldsymbol{x} \le \boldsymbol{x}^{\max} \\ & \mathbf{1}\cdot\boldsymbol{x} = 1 \\ & \mathscr{P} \text{ is a valid path from 1 to } n \end{aligned}$$

Further consider that there are two problems going on here – the smuggler's and homeland security's. In fact, we can re-write the problem above as

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & \mathbf{0} \le \boldsymbol{x} \le \boldsymbol{x}^{\max} \\ & \mathbf{1}\cdot\boldsymbol{x} = 1 \end{aligned}$$

Where

$$\begin{aligned} f(\boldsymbol{x}) = \quad \min_{\mathscr{P}} \quad & \sum_{k\in\mathscr{P}} a_j x_j \\ \text{s.t.} \quad & \mathscr{P} \text{ is a valid path from 1 to } n \end{aligned}$$

This is already good progress! Our next step is to find an expression for $f(\boldsymbol{x})$ that doesn't include that last constraint. It turns out this isn't too difficult. First, define a vector $\boldsymbol{w}$ such that $w_j = a_j x_j$. This vector simply consists of the strength of the interdiction along edge j. $f(\boldsymbol{x})$ is then simply the problem of finding the shortest path from node 1 to node n in a graph in which arc j has length w_j. This is a standard LP, and

$$\begin{aligned} f(\boldsymbol{x}) = \quad \min_{\boldsymbol{\chi}} \quad & \boldsymbol{\chi}\cdot\boldsymbol{w} \\ \text{s.t.} \quad & A\boldsymbol{\chi} = \boldsymbol{b} \\ & \boldsymbol{\chi} \ge \mathbf{0} \end{aligned}$$

where

[8] Note that since $\boldsymbol{\chi}$ is our decision variable here, it is non-trivial that the result will happen to be 0 or 1. It turns out, however, that for this kind of problem, this is always the case. This is a property of certain linear programs. See Bertsimas & Tsitsiklis for more details.

- $\boldsymbol{\chi}$ is a vector with an entry for every arc, and such that $\chi_j = 1$ if arc $j \in \mathscr{P}$ and 0 otherwise.[8]
- $\boldsymbol{b}$ is a vector that contains 1 for node 1, contains -1 for node n and contains 0 everywhere else.
- A is a matrix whose $(i,j)^{th}$ element is 1 if an arc exists from $i \to j$, equal to -1 if an arc exists from $j \to i$, and 0 otherwise.

Thus, $\boldsymbol{\chi} \cdot \boldsymbol{w} = \sum_{k \in \mathscr{P}} a_j x_j$ and the constraints $A\boldsymbol{\chi}$ ensures that $\mathscr{P}$ is indeed a valid path.

This is, once again, good progress. Our combined program now looks like

$$\begin{aligned} \max_{\boldsymbol{x}} \min_{\boldsymbol{\chi}} \quad & \boldsymbol{\chi} \cdot \boldsymbol{w} \\ \text{s.t.} \quad & \mathbf{0} \leq \boldsymbol{x} \leq \boldsymbol{x}^{\max} \\ & \mathbf{1} \cdot \boldsymbol{x} = 1 \\ & A\boldsymbol{\chi} = \boldsymbol{b} \\ & \boldsymbol{\chi} \geq \mathbf{0} \end{aligned}$$

Sadly, the objective still isn't quite right – it still involves a max min, which isn't easy to handle. To overcome this last hurdle, we will use linear programming duality to find an alternative expression for $f(\boldsymbol{x})$. By strong duality for LPs, we find that

$$\begin{aligned} f(\boldsymbol{x}) = \quad \max_{\boldsymbol{\lambda}} \quad & \boldsymbol{\lambda} \cdot \boldsymbol{b} \\ \text{s.t.} \quad & A^\top \boldsymbol{\lambda} \leq \boldsymbol{w} \end{aligned}$$

Combining everything, we therefore find that Homeland Security's program is

$$\begin{aligned} \max_{\boldsymbol{x}, \boldsymbol{\lambda}} \quad & \boldsymbol{\lambda} \cdot \boldsymbol{b} \\ \text{s.t.} \quad & \mathbf{0} \leq \boldsymbol{x} \leq \boldsymbol{x}^{\max} \\ & \mathbf{1} \cdot \boldsymbol{x} = 1 \\ & A^\top \boldsymbol{\lambda} \leq \boldsymbol{w} \end{aligned}$$

which is, indeed, an LP!

■ □ ■