CS 170 HW 7

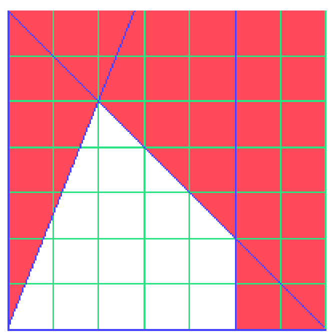
Due on 2018-03-12, at 11:59 pm

1 (★) Linear Programming Basics

Plot the feasible region and identify the optimal solution for the following linear program.

$$\text{maximize } 5x + 3y$$
$$5x - 2y \geq 0, \quad x + y \leq 7, \quad x \leq 5, \quad x \geq 0, \quad y \geq 0$$

Solution: Here is the feasible region:



The feasible region is shown in white.

We know our optimum must occur at a vertex. The vertex of (5, 2) with objective value 31 achieves the maximum value.

2 (★★★) Modeling: Tricks of the Trade

One of the most important problems in the field of *statistics* is the *linear regression problem*. Roughly speaking, this problem involves fitting a straight line to statistical data represented by points – $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ – on a graph. Denoting the line by $y = a + bx$, the objective is to choose the constants a and b to provide the "best" fit according to some criterion. The criterion usually used is the *method of least squares*, but there are other interesting criteria where linear programming can be used to solve for the optimal values of a and b. For each of the following criteria, formulate the linear programming model for this problem:

1. Minimize the sum of the absolute deviations of the data from the line; that is,

$$\min \sum_{i=1}^{n} |y_i - (a + bx_i)|$$

(*Hint:* Define a new variable $z_i = y_i - (a + bx_i)$. Notice that z_i can be either positive or negative. Any number, positive or negative, however, can be represented as the difference of two non-negative numbers. Also define as non-negative variables z_i^+ and z_i^- such that $z_i = z_i^+ - z_i^-$. How can we minimize $|z_i|$ by either minimizing or maximizing some function of z_i^+ and z_i^-?)

2. Minimize the maximum absolute deviation of the data from the line; that is,

$$\min \max_{i=1\ldots n} |y_i - (a + bx_i)|$$

(*Hint:* You'll need to start by using the same trick as above. Then consider how we can turn our objective function into a single minimization or maximization.)

Solution:

(i) Given n data points (x_i, y_i) for $i = 1, 2, \ldots n$, as well as variables a and b, define new variables $z_i = y_i - (a + bx_i)$ to denote the deviation of the i^{th} data point from the line $y = a + bx$. We know that any number, positive or negative, can be represented by the difference of two positive numbers, so that $z_i = z_i^+ - z_i^-$ and $z_i^+ \geq 0$ and $z_i^- \geq 0$ (so we are also introducing new variables z_i^+ and z_i^-).

Observe that minimizing $|z_i|$ (which is non-linear) is equivalent to minimizing $z_i^+ + z_i^-$ (which is linear) subject to the above three constraints.

Why is this the case? Let's try a small example. We could represent the number 4 as 4 - 0, 5 - 1, 6 - 2, 100 - 96, etc. But notice that the pair that gives us the smallest sum is 4 and 0. And in general for z_i, the z_i^+, z_i^- pair that gives us the smallest sum will always be either $z_i^+ = z_i$ and $z_i^- = 0$ for positive z_i, or $z_i^+ = 0$ and $z_i^- = z_i$ for negative z_i.

Hence

$$\min \sum_{i=1}^{n} |y_i - (a + bx_i)|$$

is equivalent to

$$\text{Minimize } \sum_{i=1}^{n} z_i^+ + z_i^-$$
$$\text{subject to} \begin{cases} y_i - (a + bx_i) = z_i^+ - z_i^- & \text{for } 1 \leq i \leq n \\ z_i^+ \geq 0 & \text{for } 1 \leq i \leq n \\ z_i^- \geq 0 & \text{for } 1 \leq i \leq n \end{cases}.$$

(ii) By the same reasoning, we introduce variables $z_i^+ - z_i^- = y_i - (a + bx_i)$, for $z_i^+ \geq 0$ and $z_i^- \geq 0$. Observe that minimizing $|z_i|$ (which is non-linear) is equivalent to minimizing $\max\{z_i^+, z_i^-\}$ (still non-linear) subject to the above constraints, which is equivalent to minimizing t such that $t \geq z_i^+$ and $t \geq z_i^-$ (which is linear). In simpler words, the smallest possible upper bound on z_i^+ and z_i^- will always be precisely equal to the larger of the two. Hence

$$\min \max_{i=1\ldots n} |y_i - (a + bx_i)|$$

is equivalent to

$$\text{Minimize } t$$
$$\text{subject to} \begin{cases} y_i - (a + bx_i) = z_i^+ - z_i^- & \text{for } 1 \le i \le n \\ z_i^+ \ge 0 & \text{for } 1 \le i \le n \\ z_i^- \ge 0 & \text{for } 1 \le i \le n \\ z_i^+ \le t & \text{for } 1 \le i \le n \\ z_i^- \le t & \text{for } 1 \le i \le n \end{cases} .$$

Remark: The following is an alternative solution to (i):

$$\text{Minimize } \sum_{i=1}^{n} t_i$$
$$\text{subject to} \begin{cases} y_i - (a + bx_i) \le t_i & \text{for } 1 \le i \le n \\ y_i - (a + bx_i) \ge -t_i & \text{for } 1 \le i \le n \end{cases} .$$

The following is an alternative solution to (ii):

$$\text{Minimize } t$$
$$\text{subject to} \begin{cases} y_i - (a + bx_i) \le t & \text{for } 1 \le i \le n \\ y_i - (a + bx_i) \ge -t & \text{for } 1 \le i \le n \end{cases} .$$

3 (★★) Spaceship

A spaceship is being designed to take astronauts to Mars and back. This ship will have three compartments, each with its own independent life support system. The key element in each of these life support systems is a small *oxidizer* unit that triggers a chemical process for producing oxygen. However, these units cannot be tested in advance, and only some succeed in triggering this chemical process. Therefore it is important to have several backup units for each system. Because of differing requirements for the three compartments, the units needed for each have somewhat different characteristics. A decision must now be made on just *how many* units to provide for each compartment, taking into account design limitations on the *total* amount of *space*, *weight* and *cost* that can be allocated to these units for the entire ship. The following table summarizes these limitations as well as the characteristics of the individual units for each compartment:

Compartment	Space (cu in.)	Weight (lb)	Cost ($)	Probability of failure
1	40	15	30,000	0.30
2	50	20	35,000	0.40
3	30	10	25,000	0.20
Limitation	500	200	400,000	

The objective is to *minimize the probability* of all units failing in all three compartments, subject to the above limitations and the further restriction that each compartment have a probability of no more than 0.05 that all its units fail.

Formulate the *integer programming model* for this problem. An integer programming model is the same as a linear programming model with the added functionality that variables can be forced to be integers. (*Hint:* Use logarithms.)

Integer programming is often intractable, so we use a linear program as a heuristic. We can take out the integrality constraints and change our model into a linear program. We then round the solution and make sure none of constraints have been violated (you should think about why this won't always give us the optimum)

Solution: Let x_1 be the number of oxidizer units provided for compartment 1, and define similarly x_2 for compartment 2 and x_3 for compartment 3. The probability that all units fail in compartment 1 is $(0.3)^{x_1}$, for compartment 2 is $(0.4)^{x_2}$ and for compartment 3 is $(0.2)^{x_3}$. The probability that all units fail in all compartments is $(0.3)^{x_1}(0.4)^{x_2}(0.2)^{x_3}$. The exponential function is non-linear, so we take logarithm (which preserves ordering of numbers) to get the linear programme

$$\text{Minimize } \log(0.3)x_1 + \log(0.4)x_2 + \log(0.2)x_3$$

$$\text{subject to} \begin{cases} 40x_1 + 50x_2 + 30x_3 \leq 500 & \text{space constraint (cu in.)} \\ 15x_1 + 20x_2 + 10x_3 \leq 200 & \text{weight constraint (lb)} \\ 30x_1 + 35x_2 + 25x_3 \leq 400 & \text{cost constraint (\$1000)} \\ \log(0.3)x_1, \log(0.4)x_2, \log(0.2)x_3 \leq \log(0.05) & \text{reliability constraints (log scale)} \end{cases}.$$

(Note that the reliability constraints imply the non-negativity constraints that $x_1, x_2, x_3 \geq 0$.)

4 (★★★) Generalized Max Flow

Consider the following generalization of the maximum flow problem.

You are given a directed network $G = (V, E)$ with edge capacities $\{c_e\}$. Instead of a single (s, t) pair, you are given multiple pairs $(s_1, t_1), ..., (s_k, t_k)$, where the s_i are sources of G and t_i are sinks of G. You are also given k demands $d_1, ..., d_k$. The goal is to find k flows $f^{(1)}, ..., f^{(k)}$ with the following properties:

(a) $f^{(i)}$ is a valid flow from s_i to t_i.

(b) For each edge e, the total flow $f_e^{(1)} + f_e^{(2)} + ... + f_e^{(k)}$ does not exceed the capacity c_e.

(c) The size of each flow $f^{(i)}$ is at least the demand d_i.

(d) The size of the *total* flow (the sum of the flows) is as large as possible.

Find a polynomial time algorithm to solve this generalization.

Do not give a four part solution for this problem. Main idea and runtime are sufficient.

Solution: This can be formulated as a linear program. We have variables $f^{(i)}_{(u,v)}$ for all $1 \leq i \leq k$ and $(u, v) \in E$. The total flow across each edge should be at most the capacity of the edge. Hence, we have the constraint

$$\forall (u,v) \in E \qquad \sum_{i=1}^{k} f^{(i)}_{(u,v)} \leq c_{(u,v)}$$

Also, flow conservation for the flow $f^{(i)}$ must hold for all vertices except s_i and t_i. This gives the set of constraints

$$\forall 1 \leq i \leq k, \forall v \in V \backslash \{s_i, t_i\} \qquad \sum_{(u,v)\in E} f^{(i)}_{(u,v)} = \sum_{(v,w)\in E} f^{(i)}_{(v,w)}$$

Finally, the for each i, $f^{(i)}$ must be greater than the corresponding demand.

$$\forall 1 \leq i \leq k \qquad \sum_{(s_i,u)\in E} f^{(i)}_{(s_i,u)} \geq d_i$$

The objective is simply to maximize the sum of all the flows.

$$\max \sum_{i=1}^{k} \sum_{(s_i,u)\in E} f^{(i)}_{(s_i,u)}$$

5 (★★★★) Reductions Among Flows

Show how to reduce the following variants of Max-Flow to the regular Max-Flow problem, i.e. do the following steps for each variant: Given a graph G and the additional variant constraints, show how to construct a graph G' such that

(1) If F is a flow in G satisfying the additional constraints, there is a flow F' in G' of the same size,

(2) If F' is a flow in G', then there is a flow F in G satisfying the additional constraints with the same size.

Prove that properties (1) and (2) hold for your graph G'.

1. **Max-Flow with Vertex Capacities:** In addition to edge capacities, every vertex $v \in G$ has a capacity c_v, and the flow must satisfy $\forall v : \sum_{u:(u,v)\in E} f_{uv} \leq c_v$.

2. **Max-Flow with Multiple Sources:** There are multiple source nodes $s_1, \ldots, s_k$, and the goal is to maximize the total flow coming out of all of these sources.

3. **Feasibility with Capacity Lower Bounds: (Extra Credit)** In addition to edge capacities, every edge (u, v) has a demand d_{uv}, and the flow along that edge must be at least d_{uv}. Instead of proving (1) and (2), design a graph G' and a number D such that if the maximum flow in G' is at least D, then there exists a flow in G satisfying $\forall (u, v) : d_{uv} \leq f_{uv} \leq c_{uv}$.

Solution:

1. Split every vertex v into two vertices, v_{in} and v_{out}. For each edge (u, v) with capacity c_{uv} in the original graph, create an edge (u_{out}, v_{in}) with capacity c_{uv}. Finally, if v has capacity c_v, then create an edge (v_{in}, v_{out}) with capacity c_v. If F' is a flow in this graph, then setting $F(u, v) = F'(u_{out}, v_{in})$ gives a flow in the original graph. Moreover, since the only outgoing edge from v_{in} is (v_{in}, v_{out}), and incoming flow must be equal to outgoing flow, there can be at most c_v flow passing through v. Likewise, if F is a flow in the original graph, setting $F'(u_{out}, v_{in}) = F(u, v)$, and $F'(v_{in}, v_{out}) = \sum_u F(u, v)$ gives a flow in G'. One can easily see that these flows have the same size.

2. Create one "supersource" S with edges (S, s_i) for each s_i, and set the capacity of these edges to be infinite. Then if F is a flow in G, set $F'(S, s_i) = \sum_u F(s_i, u)$. Conversely, if F' is a flow in G', just set $F(u, v) = F'(u, v)$ for $u \neq S$, and just forget about the edges from S. One can easily see that these flows have the same size.

3. Add two vertices to G, call them s' and t', and add edges (s', v) with capacity $\sum_u d(u, v)$ and (v, t') with capacity $\sum_u d(v, u)$. Add an edge (t, s) with capacity ∞, and change the capacities of all the edges in G to $c(u, v) - d(u, v)$. Let $D = \sum_{(u,v)\in E} d(u, v)$. We consider s' and t' to be the new source and sink. Note that the cuts that consist of only $\{s'\}$ or only $\{t'\}$ have value D, so D is an upper bound on the size of the max-flow.

   If G has a feasible flow F, we construct a flow F' on G' by fully saturating the edges leaving s' and coming into t', setting $F'(u, v) = F(u, v) - d(u, v)$, and $F'(t, s) = \text{size}(F)$. Since $F(u, v) \leq c(u, v)$, all the capacity constraints are satisfied, and adding up the incoming flow at a single vertex, we get

   $$\sum_{u\in G'} F'(u, v) = F'(s', v) + \sum_{u\in G} (F(u, v) - d(u, v)) = \sum_{u\in G} F(u, v)$$

   which is the incoming flow to v in F. Likewise, the outgoing flow in F' is also equal to the outgoing flow in F, and since F is a flow these must be equal, so indeed F' is a valid flow. Since all the (s', v) edges are fully saturated, this flow has size exactly D.

   To show the other direction, let G' have a flow of size exactly D. Note this implies that all the (s', v) and (v, t') edges must be fully saturated, since the corresponding cuts have value D. Set $F(u, v) = F'(u, v) + d(u, v)$. Since $F'(u, v) \leq c(u, v) - d(u, v)$, these values satisfy the capacity and demand constraints. Now adding up the incoming flow at a single (non-source or -sink) vertex,

   $$\sum_{u\in G} F(u, v) = \sum_{u\in G', u\neq s'} (F'(u, v) + d(u, v)) = \sum_{u\in G', u\neq s'} F(u, v) + F(s', v) = \sum_{u\in G'} F'(u, v)$$

   where the last equality holds because each $F(s', v)$ must be fully saturated. Thus the flow incoming into v in F is equal to the flow incoming into v in F'. A similar argument shows the outgoing flow from v in F is equal to the outgoing flow from v in F'. Since F' is a flow, these must be equal, and so F is a flow.

6 (★★★★★) A Flowy Metric

Consider an undirected graph G with capacities $c_e \geq 0$ on all edges. G has the property that any cut in G has capacity at least 1. For example, a graph with a capacity of 1 on all edges is connected if and only if all cuts have capacity at least 1. However, c_e can be an arbitrary nonnegative number in general.

1. Show that for any two vertices $s, t \in G$, the max flow from s to t is at least 1.

2. Define the *length* of a flow f to be $\texttt{length}(f) = \sum_{e \in G} |f_e|$. Define the *flow distance* $d_{\texttt{flow}}(s,t)$ to be the minimum length of any $s-t$ flow f that sends one unit of flow from s to t and satisfies all capacities; i.e. $|f_e| \leq c_e$ for all edges e.

   Show that if $c_e = 1$ for all edges e in G, then $d_{\texttt{flow}}(s,t)$ is the length of the shortest path in G from s to t.

   (*Hint*: Let $d(s,t)$ be the length of the shortest path from s to t. A good place to start might be to first try to show $d_{\texttt{flow}}(s,t) \leq d(s,t)$. Then try to show $d_{\texttt{flow}}(s,t) \geq d(s,t)$)

3. **(Extra Credit)** The shortest path satisfies the *triangle inequality*, that is for three vertices, s, t, and u in G, if $d(x,y)$ is the length of the shortest path from x to y, then $d(s,t) \leq d(s,u) + d(u,t)$. Show that the triangle inequality also holds for the flow distance. That is; show that for any three vertices $s, t, u \in G$

   $$d_{\texttt{flow}}(s,t) \leq d_{\texttt{flow}}(s,u) + d_{\texttt{flow}}(u,t)$$

   even when the capacities are arbitrary nonnegative numbers.

Solution:

1. For any two vertices s, t, the min cut separating s and t has value at least 1. Therefore, by the max flow-min cut theorem, the maximum $s-t$ flow has value at least 1.

2. First, we show that $d_{\texttt{flow}}(s,t) \leq d(s,t)$ for all $s, t \in G$. Let f be the unit $s-t$ flow that sends one unit of flow along the shortest path from s to t. Then $\texttt{length}(f)$ is the length of this path, which is $d(s,t)$. Since $d_{\texttt{flow}}(s,t)$ is the minimum length of any unit $s-t$ flow, $d_{\texttt{flow}}(s,t) \leq \texttt{length}(f) = d(s,t)$.

   Now, we show that $d(s,t) \leq d_{\texttt{flow}}(s,t)$. Consider the minimizing flow; that is the unit $s-t$ flow f with length $d_{\texttt{flow}}(s,t)$. Any unit $s-t$ flow f can be written as

   $$f = \sum_{\text{s-t paths } p \text{ in } G} w_p f_p$$

   where f_p is the unit flow along the path p, $\sum_{\text{s-t paths } p} w_p = 1$, $w_p \geq 0$ for all $s-t$ paths p, and all paths go in the same direction from s to t through any edge e. In other words, we can argue that a flow f will send some (potentially zero) fraction of its flow through each possible $s-t$ path. For example, if there are exactly 2 paths from s to t, we could send exactly half of our flow along each path, or .3 along one path and .7 along the other. Notice this is true regardless of whether or not these paths are disjoint.

This means that

$$\texttt{length}(f) = \sum_{\text{s-t paths } p \text{ in } G} w_p \texttt{length}(f_p)$$

i.e. $\texttt{length}(f)$ is the average of the lengths of many $s-t$ paths in G. In particular, there exists an $s-t$ path q in G with $\texttt{length}(f_q) \le \texttt{length}(f) = d_{\texttt{flow}}(s,t)$. $d(s,t) \le \texttt{length}(f_q)$, finishing the proof of the desired inequality.

3. Let f and g be the $s-u$ and $u-t$ flows with lengths $d_{\texttt{flow}}(s,u)$ and $d_{\texttt{flow}}(u,t)$ respectively. We wish we could define an $s-t$ flow $r = f+g$ and use this flow to show that $d_{\texttt{flow}}(s,t) \le \texttt{length}(f) + \texttt{length}(g)$. $f+g$ does not have to respect the capacities c, so it is not feasible. This is strange, though, because f and g individually do satisfy the capacities c. In order for $f+g$ to violate an edge capacity, flow from both f and g must cross the violated edge. But this implies that the $f+g$ flow would form a cycle, which is wasteful. We want to remove these cycles.

   We now formalize this intuition using max flow-min cut duality. Define a new capacity vector c' with $c'_e = \max(|f_e|, |g_e|)$. Notice that $c'_e \le c_e$ for all edges e because f and g respect G's capacities, so any c'-respecting flow is also a c-respecting flow. Futhermore, any c'-respecting flow h has length at most

$$\texttt{length}(h) \le \sum_e c'_e \le \sum_e |f_e| + |g_e| = \texttt{length}(f) + \texttt{length}(g) = d_{\texttt{flow}}(s,u) + d_{\texttt{flow}}(u,t)$$

   Therefore, it suffices to show that there is a unit $s-t$ c'-respecting flow. By max flow-min cut, it suffices to show that all $s-t$ cuts have capacity at least 1. Consider any set of vertices S containing s but not t. If $u \notin S$, then $1 \le \sum_{e \in \partial S} |f_e| \le \sum_{e \in \partial S} c'_e$. If $u \in S$, then $1 \le \sum_{e \in \partial S} |g_e| \le \sum_{e \in \partial S} c'_e$. In particular, the cut defined by S always has capacity at least 1. Therefore, the min $s-t$ c'-cut has capacity at least 1. Therefore, there exists a unit $s-t$ flow with length at most $d_{\texttt{flow}}(s,u) + d_{\texttt{flow}}(u,t)$, as desired.