ISyE/CS 719: Stochastic Programming
Fall 2016
Assignment #1

Example Solutions

1. Show that the expected value of perfect information is nonnegative, i.e.,

$$\min_{x\in X} \mathbb{E}[F(x,\xi))] - \mathbb{E}[\min_{x\in X} F(x,\xi))] \geq 0.$$

(You may do this for the case of a discrete random variable having $\mathbb{P}(\xi = \xi^k) = p_k, k = 1, \ldots, K$, where $p \geq 0$ and $\sum_{k=1}^K p_k = 1$.)

Answer:
Let x^* be an optimal value to the problem $\min_{x\in X} \mathbb{E}[F(x,\xi)]$. Then, for each scenario ξ, as x^* is one possible solution to the problem $\min_{x\in X} F(x,\xi)$, it holds that

$$\min_{x\in X} F(x,\xi) \leq F(x^*,\xi).$$

Thus,

$$\mathbb{E}[\min_{x\in X} F(x,\xi))] = \sum_{k=1}^K p_k \min_{x\in X} F(x,\xi^k) \leq \sum_{k=1}^K p_k F(x^*,\xi^k) = \min_{x\in X} \mathbb{E}[F(x,\xi))]$$

where the last equation follows by the definition of x^* as an optimal solution to the last problem. ◊

2. Let V be a finite set of vectors in $\mathbb{R}^m$. Let h be a vector in $\mathbb{R}^m$ and let T be an $m \times n$ matrix. Let

$$Q(x) \overset{\text{def}}{=} \max\{\pi^\top(h - Tx) : \pi \in V\}$$

for $x \in X$, where $X \subseteq \mathbb{R}^n$ is a convex set. Show that Q is convex over $x \in X$.

Answer:
Let $x^1, x^2 \in X$ and $\lambda \in (0,1)$. Then

$$\begin{aligned} Q(\lambda x^1 + (1-\lambda)x^2) &= \max\{\pi^\top(h - T\lambda x^1 - T\lambda x^2) : \pi \in V\} \\ &= \max\{\lambda\pi^\top(h - Tx^1) + (1-\lambda)\pi^\top(h - Tx^2) : \pi \in V\} && (1) \\ &\leq \max\{\lambda\pi^\top(h - Tx^1) : \pi \in V\} + \max\{(1-\lambda)\pi^\top(h - Tx^1) : \pi \in V\} && (2) \\ &= \lambda Q(x^1) + (1-\lambda)Q(x^2) && (3) \end{aligned}$$

which is the definition of convexity. The inequality in (2) follows because if π^* is a vector that achieves the maximum in (1), then π^* is one option in each of hte two different max operations in (2), which would yield the same value as in (1) (but a larger value in (2) is possible using different π values). ◊

3. Consider the following two-stage stochastic linear program

$$z^{SP} = \min_{x\in X} \sum_{k=1}^{K} p_k h(x, d^k)$$

where $X = \{x : Ax = b, x \geq 0\}$ is bounded and non-empty and

$$\begin{aligned} h(x, d) = \quad \min_{y} \quad & q^\top y \\ \text{s.t.} \quad & Wy = d - Tx \\ & y \geq 0. \end{aligned}$$

Thus d varies by scenario k (i.e., it is random), while the data q, W and T are fixed. The "perfect information" objective is defined as

$$z^*_{PI} = \sum_{k=1}^{K} p_k \min_{x\in X} h(x, d^k)$$

and the "mean-value" objective is defined as

$$z^*_D = \min_{x\in X} h(x, \bar{d}).$$

where $\bar{d} = \sum_{k=1}^{K} p_k d^k$. Assume the two-stage stochastic linear program has relatively complete recourse, and $\{\pi : \pi W \leq q\} \neq \emptyset$. Show

$$z^*_{PI} \geq z^*_D.$$

(Note: In general, this result does not hold if parameters in addition to d are random.)

Answer:
First, under the assumptions we can apply strong duality to conclude

$$h(x, d) = \max\{\pi^\top (d - Tx) : \pi^\top W \leq q\}.$$

Next, observe that this implies that $h(x, d)$ is *jointly convex* in (x, d) because it is a maximum of affine functions in (x, d). (Note that here it is critical that T is fixed.) Now, for each scenario $k = 1, \dots, K$, let x^k be an optimal solution to the problem

$$\min_{x\in X} h(x, d^k)$$

and let $\bar{x} = \sum_{k=1}^{k} p_k x^k$. By convexity of the function $h(x, d)$ and Jensen's inequality, it follows that

$$h(\bar{x}, \bar{d}) \leq \sum_{k=1}^{K} p_k h(x^k, d^k) = \sum_{k=1}^{K} \min_{x\in X} h(x, d^k) = z^*_{PI}.$$

The proof is then concluded by observing that convexity of X implies that $\bar{x}$ is a feasible solution to the problem $\min_{x\in X} h(x, \bar{d})$ and hence

$$\min_{x\in X} h(x, \bar{d}) \leq h(\bar{x}, \bar{d}).$$

$\Diamond$

4. Let $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ be a polyhedron. Show that r is a ray of P if and only if $r \in \text{recc}(P)$ (i.e., $Ar \leq 0$).

   Answer:
   First assume r satisfies $Ar \leq 0$. Let $x \in P$ and $\lambda > 0$. Then

$$A(x + \lambda r) = Ax + \lambda Ar \leq Ax \leq b.$$

   Thus, $x + \lambda r \in P$ for all $\lambda > 0$, and hence r is a ray of P.

   Next, we show that if r is a ray then $Ar \leq 0$ by showing the contrapositive: if $Ar \not\leq 0$, then r is not a ray. Thus, assume that for some row a_i of A, it holds that $a_i^\top r > 0$. Let $x \in P$ and $\lambda > 0$. Then, as $\lambda \to \infty$,

$$a_i^\top (x + \lambda r) = a_i^\top x + \lambda a_i^\top r \to +\infty$$

   and hence the inequalities $A(x+\lambda r) \leq b$ cannot be satisfied for all $\lambda > 0$, and so r is not a ray. $\Diamond$

The following problem description applies for all problems 5-8.

The Jeffrey Lebowski White Russian Company (JLWRC) ships white russians across the country. The supply chain for JLWRC consists of a set F of production facilities. Each production facility $i \in F$ has a maximum production capacity of b_i units. From the production facilities, the white russian is sent to a set H of transportation hubs, and finally from the transportation hubs, the white russian is sent to a set of customers C. The set of transportation arcs in the network is $A = (F \times H) \cup (H \times C)$, and each arc $a \in A$ has an existing capacity u_a.

Jeffrey Lebowski (The Dude) wants to increase the capacity of his transportation network in order to "best" meet an unknown demand from his customers. There is a set of (equally likely) demand scenarios $k = 1, \ldots, K$, and each customer $j \in C$ would like to receive d_j^k units of White Russian in scenario k. The cost of increasing the capacity of arc a by 1 unit is given by the parameter c_a.

Demand must be met in every scenario, and in case the Dude cannot meet demand, he must buy the extra White Russian for customer $j \in C$ in scenario $k = 1, \ldots, K$ from Jesus Quintana at a cost of f_j/unit.

5. Write the extensive form of a two-stage stochastic program that minimizes the cost of installation of capacity plus the expected cost of meeting demand. Does this model have complete recourse? Does it have relatively complete recourse?

   Answer:
   First-stage decision variables: x_{ij} = capacity to add to arc $(i, j) \in A$

   Second-stage decision variables:

   - y_{ijk} = flow on arc (i, j) in scenario k
   - z_{jk} = unmet demand at customer $j \in C$ in scenario k

   Objective:

$$\min \sum_{(i,j) \in A} c_{ij} x_{ij} + \frac{1}{K} \sum_{k=1}^{K} \sum_{j \in C} f_j z_{jk}$$

Constraints:

$$\begin{aligned}
y_{ijk} &\leq u_{ij} + x_{ij}, && (i,j) \in A, k = 1, \ldots, K \\
\sum_{j \in H} y_{ijk} &\leq b_i, && i \in F, k = 1, \ldots, K \\
\sum_{i \in F} y_{ijk} - \sum_{\ell \in C} y_{j\ell k} &= 0, && j \in H, k = 1, \ldots, K \\
\sum_{i \in H} y_{ijk} + z_{jk} &= d_j^k, && j \in C, k = 1, \ldots, K \\
y_{ijk}, z_{jk} &\geq 0, x_{ij} \geq 0
\end{aligned}$$

The first constraint specifies that the flow on an arc in any scenario should not exceed the current plus installed capacity. The second constraint limits the production from each facility in each scenario. The third constraint is flow balance (White Russian in equals White Russian out) for each hub in each scenario. The fourth constraint indicates that in each scenario the total of White Russian supplied to each customer, including that purchased from Jesus Quentana, must equal the demand in each scenario.

This model **does not** have complete recourse, because if for some $(i,j) \in A$, $x_{ij} < -u_{ij}$, then the first constraint cannot be satisfied since $y_{ijk} \geq 0$. The model **does** have relatively complete recourse. This can be seen because it is always feasible to set $y_{ijk} = 0$ for all arcs (i,j) and to set $z_{jk} = d_j^k$ for all customers j. ◊

6. Implement the model from problem 5, and use it to solve the two instance files on the course web site. The course web site also contains a "starter" python file that will read the data from the file you select. Assume that all scenarios are equally likely. Also evaluate the Value of the Stochastic Solution. Include in your hard-copy submission a printout of your code, the optimal solution value for the two instances, the solution times for the two instances, and the value of the stochastic solution. Also submit an electronic copy of your code in the Dropbox on the Learn@UW course web site.

Answer:
See the posted example solution file on the course web site. The table below summarizes the solution value, value of stochastic solution (VSS), and solution time for each instance.

	ObjValue	VSS	Total time (sec)	Gurobi time
nd10-4-10-15.pdat	5933.71	1542.39	0.06	0.01
nd15-10-20-3000.pdat	7421.84	2762.95	236.3	195.9

Note that I reported two values for solution time (in seconds), the "Total time" and "Gurobi time". "Total time" is determined using python's timing functionality, and includes also the time to build the model. "Gurobi time" is the solution time reported by Gurobi for solving the model. For large models, the building time can be significant. Also, I did not restrict the number of threads Gurobi could use. Gurobi used 3 threads (on my MacBook laptop), and the time reported is wall clock time.

◊

7. Derive the form of a Benders cut (for a multi-cut implementation), for the model in problem 5. (I.e., give the form of the cut as an explicit function of the dual variables.)

   Answer:
   The scenario k subproblem has the form:

$$
\begin{aligned}
Q_k(x) = \min \ & \sum_{j \in C} f_j z_j \\
\text{s.t.} \ & -y_{ij} \geq -u_{ij} - x_{ij}, && (i,j) \in A \ (\pi^A_{ij}) \\
& -\sum_{j \in H} y_{ij} \geq -b_i, && i \in F \ (\pi^F_i) \\
& \sum_{i \in F} y_{ij} - \sum_{\ell \in C} y_{j\ell} = 0, && j \in H, (\pi^H_j) \\
& \sum_{i \in H} y_{ij} + z_j = d^k_j, && j \in C (\pi^C_j) \\
& y_{ij}, z_j \geq 0
\end{aligned}
$$

   Using the dual variable notation labeled in parentheses next to the constraints above, we can write down the dual of this linear program as:

$$
\begin{aligned}
\max \ & \sum_{j \in C} \pi^C_j d^k_j - \sum_{i \in F} \pi^F_i b_i - \sum_{(i,j) \in A} \pi^A_{ij} (u_{ij} + x_{ij}) \\
\text{s.t.} \ & \pi^H_j - \pi^F_i - \pi^A_{ij} \leq 0, && (i,j) \in F \times H \\
& -\pi^H_i + \pi^C_j - \pi^A_{ij} \leq 0, && (i,j) \in H \times C \\
& \sum_{j \in C} \pi^C_j \leq f_j, && j \in C \\
& \pi^A_{ij} \geq 0, \pi^F_i \geq 0, \pi^H_j \text{ free}, \pi^C_j \text{ free}
\end{aligned}
$$

   Note: For deriving the form of the Benders cut, the important part of the dual is just the objective function.

   Therefore, if $\hat{\pi}$ is an extreme point solution of the constraints listed above, then the Benders cut has the form:

$$
\theta_k \geq \Big(\sum_{j \in C} \hat{\pi}^C_j d^k_j - \sum_{i \in F} \hat{\pi}^F_i b_i - \sum_{(i,j) \in A} \hat{\pi}^A_{ij} u_{ij} \Big) - \sum_{(i,j) \in A} \hat{\pi}^A_{ij} x_{ij}.
$$

   Note: The signs I obtained are due to writing the primal form of the subproblem in "standard form", so all dual variables are either ≥ 0 or free. An alternative correct solution can be obtained by writing the subproblem differently (e.g., $y_{ij} \leq u_{ij} + x_{ij}$) in which case the corresponding dual variable would have the sign restriction $\pi^A_{ij} \leq 0$, and the coefficient in the dual objective would have the sign flipped. (This is also relevant for the implementation, in which the signs may have to be flipped depending on how the constraint was written when building the model.) ◊

8. Implement a Benders cutting plane algorithm to solve the two instances from problem 6. Have your algorithm display the lower and upper bound obtained after each iteration, and number

of cuts added at each iteratoin. Submit a printout of this output and the code with your hard-copy submission. Also submit the code in the Dropbox on the Learn@UW course web site.

Answer:
See the implementation file posted on the course web site. The table below indicates the number of iterations and total time of the algorithm for solving the two instances.

	Total time (sec)	Iterations
nd10-4-10-15.pdat	0.14	10
nd15-10-20-3000.pdat	141.2	12

This is slightly faster than solving the extensive form, but I would not consider the difference in this case to be worth the extra implementation effort.

I also did some profiling to investigate how much time was spent in different parts of the algorithm for the larger instance. I found that the time to solve the master problem was very small, so the time in the algorithm is spent mostly building and solving the scenario subproblems (something that is very easy to parallelize), and building the cuts and adding them to the master problem. In addition, the time spent actually solving the subproblems was about half the time spent updating the subproblems and building and adding cuts to the master problem. (In particular, the time to do an iteration is much less in later iterations because many fewer cuts were built and added to the master problem.)

Because solving the master problem was not a significant part of the solution time, a single-cut implementation would likely not be more efficient (time for solving the subproblems in each iteration would be the same in the single-cut implementation, and one would expect many more iterations would be required).

Finally, note that in this case the number of iterations used by this basic version of Benders decomposition is very small, so it leaves little room for improvement using the cut selection ideas presented at the end of the Benders slides, or by using any of the regularization methods.

$\Diamond$