INTRODUCTION TO STOCHASTIC LINEAR PROGRAMMING

FARAZ W. RAHMAN

December 11, 2012

Abstract. We introduce the notion of stochastic linear programming, and discuss ways to deal with uncertainty in the parameters of linear programs. We concentrate primarily on the recourse approach, and describe an application in the context of the Oil Problem with uncertain levels of demand.

In the linear programs we have seen thus far, of the form

$$\begin{aligned} \min \quad & \vec{c}^T \vec{x} \\ A\vec{x} \quad & \geq \vec{b} \\ \vec{x} \quad & \geq \vec{0} \end{aligned} \tag{1}$$

we have assumed that the parameters of the problem, A, $\vec{b}$ and $\vec{c}$, are constant and that we know these values. However, in many cases some or all of these parameters will either be unknown at the time when we make our decisions for $\vec{x}$, or will be random and follow some joint distribution.

Here, we explore some of the approaches to dealing with these uncertainties, within the context of the Oil Problem. Recall, then, that the Oil Problem could be summarized as follows.

We have R refineries, each with a given maximum capacity s_i, $i \in \{1, 2, \ldots, R\}$, and C cities, each with a given level of demand for oil $d_j, j \in \{1, 2, \ldots, C\}$. We are given costs of shipping oil from each refinery i to each city j, $c_{i,j}$, which we express as a vector $\vec{c}$. Our goal is to determine quantities of oil to ship $x_{i,j}$ from each refinery i to each city j, which we summarize as the vector $\vec{x}$, in order to minimize the total shipping cost $\vec{c}^T\vec{x}$, while meeting demand in all of the cities and not exceeding capacity in any of the refineries.

As an example, for a situation with 2 refineries and 3 cities, we could express the above in the standard form as in (1) by setting

$$A = \begin{pmatrix} -1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & -1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

$$\vec{x} = \begin{pmatrix} x_{1,1} \\ x_{1,2} \\ x_{1,3} \\ x_{2,1} \\ x_{2,2} \\ x_{2,3} \end{pmatrix}, \vec{b} = \begin{pmatrix} -s_1 \\ -s_2 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix}, \vec{c} = \begin{pmatrix} c_{1,1} \\ c_{1,2} \\ c_{1,3} \\ c_{2,1} \\ c_{2,2} \\ c_{2,3} \end{pmatrix} \tag{2}$$

and solving the problem as in (1). We can refer to this as the Deterministic Problem, since all parameter values are known in advance.

We can now explore an example of stochastic linear programming, by considering the above problem with stochastic, or in other words, random, demand. We suppose that demand is not known at the time we must make our decision about quantities $\vec{x}$ to ship, but that the demand levels (d_1, d_2, d_3) follow some known joint distribution. We make another simplifying assumption, that the vector of demands can only take on some finite number of values.

From (2), we can see that in this particular case, only the vector $\vec{b}$ is stochastic, since some of its elements are random, while the other parameters of the problem, A and $\vec{c}$, are known in advance. By assumption, we know that $\vec{b}$ takes on some finite number of values $\vec{b}_s, s \in \{1, 2, \ldots, S\}$, with each $\vec{b}_s$ occurring with a known probability p_s, such that $\sum_{s=1}^{S} p_s = 1$.

We note that it is possible that the other parameters of the problem, A and $\vec{c}$, may also be stochastic in nature. In this case, we make the same assumptions, that $(A, \vec{b}, \vec{c})$ follow some joint distribution, and take on some finite number of values $(A_s, \vec{b}_s, \vec{c}_s), s \in \{1, 2, \ldots, S\}$.

We consider a simple example. Suppose we have two possible scenarios, such that $\vec{c}^T = (20, 20, 10, 50, 10, 15)$, and A is as shown in (2); however, suppose there are two possible values for $\vec{b}$:

$$\begin{aligned} &\text{Scenario 1: } \vec{b}_1 = (-300, -200, 300, 100, 100)^T; && p(\vec{b}_1) = \frac{2}{3} \\ &\text{Scenario 2: } \vec{b}_2 = (-300, -200, 200, 250, 50)^T; && p(\vec{b}_2) = \frac{1}{3} \end{aligned} \tag{3}$$

(where $p(\vec{b}_i)$ is the probability that $\vec{b}$ takes on the value $\vec{b}_i$).

We can now start to discuss various ways to address the uncertainty in this problem.

1. Expected Value approach

In the expected value approach, we compute the following:

$$\begin{aligned} A_E &= \sum_{s=1}^{S} p_s A_s \\ \vec{b}_E &= \sum_{s=1}^{S} p_s \vec{b}_s \\ \vec{c}_E &= \sum_{s=1}^{S} p_s \vec{c}_s \end{aligned} \tag{4}$$

and then solve

$$\begin{aligned} \min\ & c_E^T \vec{x} \\ & A_E \vec{x} \geq \vec{b_E} \\ & \vec{x} \geq \vec{0} \end{aligned} \tag{5}$$

This is a fairly straightforward approach, in the sense that the complexity of the problem remains unchanged, since the dimensions of the parameters remain unchanged when we compute their expected values.

However, this approach can be problematic, since its optimal $\vec{x}$ may not only be sub-optimal in some (or all) scenarios, but may in fact not be feasible in some (or any) of the individual scenarios.

Consider the two scenarios described as in (3): we obtain $\vec{b}_E = (-300, -200, 266\frac{2}{3}, 150, 83\frac{1}{3})^T$, while A_E is the same as in (2), and the expected cost vector $\vec{c}_E^T = (20, 20, 10, 50, 10, 15)$ similarly remains unchanged since the costs were determined in advance.

Now, solving the problem as in (5), we obtain the solution $\vec{x}^* = (266\frac{2}{3}, 0, 33\frac{1}{3}, 0, 150, 50)$. We note that $A\vec{x}^* = \vec{b}_E$. Now, we see that if Scenario 1 were to occur, we would have failed to meet the demand in city 1, and if Scenario 2 were to occur, we would have failed to meet demand in city 2. This would imply that the solutions we obtain from the expected value problem may not be feasible in *any* scenario.

2. Recourse Approach

The recourse approach is a more popular approach that the expected value approach, and has many applications [1]. We will first describe the recourse approach in the context of the same Oil Problem as above, and show how we can generalize it further to situations where all the parameters of the problem might be random.

Suppose that demand is unknown, and we have the opportunity to ship quantities of oil $\vec{x}$ now, at costs $\vec{c}$. Then, after demand, and hence $\vec{b}_s$, becomes known, we may need to ship additional quantities of oil $\vec{y}_s$, at scenario-specific "recourse" costs $\vec{r}_s$, in order to meet demand in all the cities; it would be realistic in many situations to have $\vec{r}_s > \vec{c}$ for all $s \in \{1, 2, \ldots, S\}$.

The constraints for this recourse problem, as in the original Oil Problem, can be divided into supply and demand constraints. Firstly, in any scenario s, the total amount shipped out of a refinery over the initial and recourse stages must not exceed the capacity of the refinery. Secondly, in any scenario, the total amount shipped into a city over the initial and recourse stages must not fall short of demand in that scenario. Using the notation in (2), we can express this as

$$A\vec{x} + A\vec{y}_s \geq \vec{b}_s$$

for each scenario $s \in \{1, 2, \ldots, S\}$.

A common objective function to use in this situation is to minimize the expected total cost. For any scenario s, the total cost we would incur is $\vec{c}^T\vec{x} + \vec{r}_s^T\vec{y}_s$. Therefore, the expected total cost can be expressed as

$$\begin{aligned}\sum_{s=1}^{S} p_s(\vec{c}^T\vec{x} + \vec{r}_s^T\vec{y}_s) &= \vec{c}^T\vec{x} + \sum_{s=1}^{S} p_s\vec{r}_s^T\vec{y}_s \\ &= \begin{pmatrix}\vec{c}^T & p_1\vec{r}_1^T & \ldots & p_S\vec{r_S}^T\end{pmatrix}\begin{pmatrix}\vec{x} \\ \vec{y_1} \\ \vdots \\ \vec{y}_S\end{pmatrix}\end{aligned}$$

and hence, we obtain as the cost vector $\begin{pmatrix}\vec{c}^T & \vec{r}_1^T & \ldots & \vec{r_S}^T\end{pmatrix}$.

Suppose, for the Oil Problem we have discussed, we have as recourse costs $\vec{r_1}^T = 2\vec{c}^T$ and $\vec{r_2}^T = 3\vec{c}^T$. We can summarize the recourse problem in block matrix form as

$$\begin{aligned} &\min \begin{pmatrix} \vec{c}^T & p_1\vec{r_1}^T & p_2\vec{r_2}^T \end{pmatrix} \begin{pmatrix} \vec{x} \\ \vec{y_1} \\ \vec{y_2} \end{pmatrix} \\ &\begin{pmatrix} A & A & \mathbf{0} \\ A & \mathbf{0} & A \end{pmatrix} \begin{pmatrix} \vec{x} \\ \vec{y_1} \\ \vec{y_2} \end{pmatrix} \geq \begin{pmatrix} \vec{b}_1 \\ \vec{b}_2 \end{pmatrix}; \end{aligned} \tag{6}$$

where $\mathbf{0}$ is a matrix of zeros of the same dimensions as A. A more extensive, and less concise, description of this problem is provided in Section 5.1. This problem is of the same form as in (1), and we obtain as a solution $\vec{x} = (200, 0, 0, 0, 100, 50)^T, \vec{y_1} = (100, 0, 0, 0, 0, 50)^T, \vec{y_2} = (0, 100, 0, 0, 50, 0)^T$.

We could refine the problem further by incorporating storage costs in the objective function. Consider that the vector $A\vec{x} + A\vec{y}_s - \vec{b}_s$ will have, in the first two entries, the quantities of oil remaining in each of the two refineries, and will have for the remaining three entries the quantities of oil in excess of demand that has been shipped to each of the three cities, in scenario s; here, the vectors and matrices are of the same form as shown in (2). Now suppose for the vector of storage costs in scenario s, we have $\vec{t}_s = (t_{1,s}, \ldots, t_{5,s})^T$, where the first two entries are the storage costs for each of the refineries, and the next three are the storage costs in each of the cities. Then, we can express the total cost in scenario s as

$$\begin{aligned} &\begin{pmatrix} \vec{c}^T & \vec{r_1}^T & \vec{r_2}^T \end{pmatrix} \begin{pmatrix} \vec{x} \\ \vec{y_1} \\ \vec{y_2} \end{pmatrix} + \begin{pmatrix} \vec{t_1}^T & \vec{t_2}^T \end{pmatrix} \left[\begin{pmatrix} A & A & \mathbf{0} \\ A & \mathbf{0} & A \end{pmatrix} \begin{pmatrix} \vec{x} \\ \vec{y_1} \\ \vec{y_2} \end{pmatrix} - \begin{pmatrix} \vec{b}_1 \\ \vec{b}_2 \end{pmatrix} \right] \\ &= \left[\begin{pmatrix} \vec{c}^T & \vec{r_1}^T & \vec{r_2}^T \end{pmatrix} + \begin{pmatrix} \vec{t_1}^T & \vec{t_2}^T \end{pmatrix} \begin{pmatrix} A & A & \mathbf{0} \\ A & \mathbf{0} & A \end{pmatrix} \right] \begin{pmatrix} \vec{x} \\ \vec{y_1} \\ \vec{y_2} \end{pmatrix} + \begin{pmatrix} \vec{t_1}^T & \vec{t_2}^T \end{pmatrix} \begin{pmatrix} \vec{b}_1 \\ \vec{b}_2 \end{pmatrix} \\ &= \left[\begin{pmatrix} \vec{c}^T & \vec{r_1}^T & \vec{r_2}^T \end{pmatrix} + \begin{pmatrix} \vec{t_1}^T & \vec{t_2}^T \end{pmatrix} \begin{pmatrix} A & A & \mathbf{0} \\ A & \mathbf{0} & A \end{pmatrix} \right] \begin{pmatrix} \vec{x} \\ \vec{y_1} \\ \vec{y_2} \end{pmatrix} + C \end{aligned}$$

where C is constant, and can be disregarded in the objective function. Including storage costs in the example we have discussed leaves the results unchanged, since in each scenario the total supply and total demand are equal, and there is consequently no opportunity to store any extra oil in any of the cities or refineries. For an example in which total supply across the refineries exceeds total demand across the cities, and hence the inclusion of storage costs can potentially alter the results of the algorithm, see Section 5.2 of the Appendix.

Note, however, that the problem described in (6) has a much higher dimensionality than that described in (2). We started with RC variables and $R + C$ constraints for the deterministic problem. If we consider S different scenarios, we end up with $(R+C)S$ constraints, and RC first stage variables, plus $S(RC)$ second stage variables, giving a total of $RC(1+S)$ variables. Therefore, we see that the problem will increase in complexity as we consider more scenarios.

In the more general case of recourse problems, each scenario would be comprised not only of a different $\vec{b}_s$, but of a set of parameters $(A_s, B_s, \vec{b}_s, \vec{c}_s, \vec{r}_s)$, with $s \in \{0, 1, \ldots, S\}$, such that we can express the recourse problem as

$$\begin{aligned} &\min \begin{pmatrix} p_1(\vec{c}_1^T + \vec{r}_1^T) & \ldots & p_S(c_S^T + \vec{r}_S^T) \end{pmatrix} \begin{pmatrix} \vec{x} \\ \vec{y_1} \\ \vdots \\ \vec{y}_S \end{pmatrix} \\ &\text{subject to} \\ &\begin{pmatrix} A_0 & \ldots & \ldots \\ A_1 & B_1 & \ldots \\ \vdots & & \ddots \\ A_S & \ldots & B_S \end{pmatrix} \begin{pmatrix} \vec{x} \\ \vec{y_1} \\ \vdots \\ \vec{y}_S \end{pmatrix} \geq \begin{pmatrix} \vec{b}_0 \\ \vec{b}_1 \\ \vdots \\ \vec{b}_S \end{pmatrix} \end{aligned} \tag{7}$$

2.1. **Multistage Recourse.** In many applications, it is likely that we will have to make decisions at multiple stages in the future, as new information is revealed to us. For example, we might extend the Oil Problem to a case where we might be supplying oil to a set of cities each month, subject to unknown levels of demand, and possibly also of supply. Here, we discuss a simple case where we have three stages. For the sake of clarity, we will refer to Figure 1, which illustrates the possible sequences of scenarios.

At each stage, the decision vector we need to pick depends on the decisions we made leading up to that stage. Each possible scenario at each stage will, therefore, comprise a set of parameters that define the relationship between that decision, and the decisions preceding it, as well as some set of costs associated with that decision. We can, therefore, express the general three-stage problem in the following form:

$$\min \begin{pmatrix} \vec{c}_1^T & (p_1 + p_2 + p_3)\vec{c}_2^T & p_4\vec{c}_3^T & p_1\vec{c}_4^T & p_2\vec{c}_5^T & p_3\vec{c}_6^T & p_4\vec{c}_7^T \end{pmatrix} \begin{pmatrix} \vec{y}_1 \\ \vec{y}_2 \\ \vec{y}_3 \\ \vec{y}_4 \\ \vec{y}_5 \\ \vec{y}_6 \\ \vec{y}_7 \end{pmatrix}$$

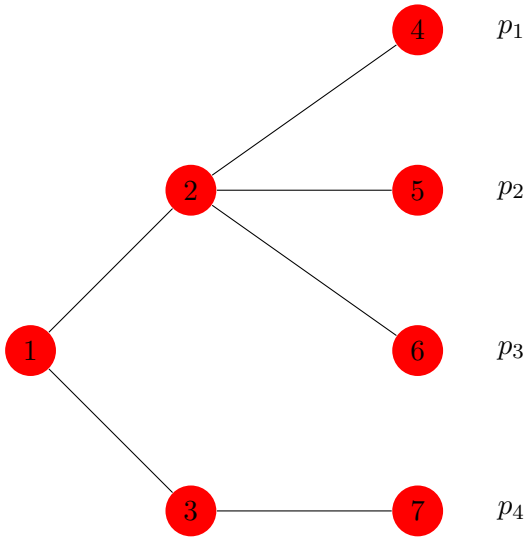

FIGURE 1. A simple scenario tree with three stages. Each p_i refers to the probability of that sequence occurring.

subject to

$$\begin{pmatrix} A_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ A_2 & B_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ A_3 & \mathbf{0} & B_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ A_4 & B_4 & \mathbf{0} & C_4 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ A_5 & B_5 & \mathbf{0} & \mathbf{0} & C_5 & \mathbf{0} & \mathbf{0} \\ A_6 & B_6 & \mathbf{0} & \mathbf{0} & \mathbf{0} & C_6 & \mathbf{0} \\ A_7 & \mathbf{0} & B_7 & \mathbf{0} & \mathbf{0} & \mathbf{0} & C_7 \end{pmatrix} \begin{pmatrix} \vec{y}_1 \\ \vec{y}_2 \\ \vec{y}_3 \\ \vec{y}_4 \\ \vec{y}_5 \\ \vec{y}_6 \\ \vec{y}_7 \end{pmatrix} \geq \begin{pmatrix} \vec{b}_1 \\ \vec{b}_2 \\ \vec{b}_3 \\ \vec{b}_4 \\ \vec{b}_5 \\ \vec{b}_6 \\ \vec{b}_7 \end{pmatrix}$$

2.2. **Cost of Perfect Information.** A metric that is sometimes taken into consideration in problems with uncertain parameters is known as the **Cost of Perfect Information**. We can think about this in the following way: supposing we knew in advance the value of all our parameters (in the case of the Oil Problem, the demand in each city) before we made the first-stage decisions. Then, we would have a deterministic linear program of the type shown in (1), where we choose $\vec{x}$ to minimize total cost $\vec{c}^T\vec{x}$. Since the recourse costs $\vec{r}_s$ will generally be higher than the first stage costs $\vec{c}$, it follows that knowing the demand levels beforehand would yield a minimum cost at least as low as when we used the recourse

approach.

In order to measure the cost of perfect information, we first compute the optimal costs $c_s^* = \vec{c}^T \vec{x}_s$ for the linear programming problems

$$\begin{aligned} \min \ \vec{c}^T \vec{x}_s \\ A\vec{x}_s \geq \vec{b}_s \\ \vec{x}_s \geq \vec{0} \end{aligned}$$

for $s = 1, 2, \ldots, S$.

We then consider the average of these values, weighted by the probability of each scenario being realized, and hence obtain the expected shipping cost, assuming we could know demand beforehand. In other words, we compute

$$\sum_{s=1}^{S} p_s c_s^*.$$

Subtracting from this the optimal total expected cost c_R^* from the two-stage recourse problem shown in (6), we calculate the cost of perfect information to be

$$c_R^* - \sum_{s=1}^{S} p_s c_s^*.$$

Using the example of the Oil Problem, we could compute the minimal cost for scenario 1 as \$8500 and for scenario 2 as \$7500, giving us an expected cost of \$8166.67. Subtracting this from the expected total cost of \$11916.67, which we obtained from solving (6), we compute, for the cost of perfect information, \$11916.67 − \$8166.67 = \$3750. These computations are shown in Section 5.2.

This metric would give us an idea of the cost we incur by having to hedge against different scenarios occurring. While in many applications, the parameter values are not only unknown but unknowable, in others the parameter values may be known to other parties, but are revealed only at a certain time, after which we can make the recourse decisions. The cost of perfect information would give us a measure of how much we might be willing to pay for knowledge of the parameter values (demand, in this case) before we make our first-stage decisions.

It is possible to enhance this idea to consider, for example, the value of market research which might, rather than give us the true parameter values in advance, at least *narrow* the possibilities for the parameter values, giving us fewer scenarios to consider in the problem.

2.3. **Alternative Objective Functions.** While the standard objective function in recourse problems is the expected total cost, it is certainly possible to consider other objective functions that might be more relevant to specific applications. One possibility is to minimize the maximum total cost that could occur in any scenario:

$$T_{\max} = \max_s \{\vec{c}^T \vec{x} + \vec{r}_s^T \vec{y}_s\}.$$

Then, we would seek to minimize $T_{\max}$ in the linear programming problem.

This approach gives us an upper bound on the total cost, and may be more applicable than the expected value approach in situations where the tolerance for risk is low.

2.4. **Network flows.** Another possible refinement of the Oil Problem would be to allow oil to be shipped between cities. In this case, we would not explicitly differentiate between refineries and cities, and would instead consider these to be N nodes in a network, rather than R refineries and C cities. We would require the $\frac{1}{2}N(N-1)$ costs of shipping between any two nodes, and would need to solve for the $\frac{1}{2}N(N-1)$ quantities to ship between any two nodes.

We can modify our constraints slightly to implicitly differentiate between nodes and cities. The idea is to include initial supplies S_i and demands D_i in each node i in the network. We would, then, set initial demand to 0 for refineries, and set initial supply to 0 in cities.

In this case, the supply and demand constraints can be summarized as **conservation of flow**:

$$\sum_{i=1}^{N} x_{j,i} \leq S_j + \sum_{i=1}^{N} x_{i,j} - D_j \text{ , for } j = 1, 2, \ldots, N$$

The above implies that the total amount shipped out of node j cannot exceed the initial supply at node j, plus the total shipped into node j, minus the demand in node j.

For the deterministic Oil Problem, this approach is effectively no different from the earlier definition of the constraints, in which we differentiated between refineries and cities explicitly. However, for the Oil Problem with stochastic demand, we could potentially find a lower optimal if we allow oil to be shipped between cities. One example to consider would be if we had two cities j and k with low shipping costs between them, but with high storage costs in each. Suppose that total demand across the two cities is always constant in all scenarios, so that $D_{j,s} + D_{k,s} = C$ for $s = 1, 2, \ldots, S$; we may find that the cost is lower when we allow oil to be shipped between cities, since we could ship oil from the low demand city to the high demand city at a cost which is lower than storing in the low demand city. Suppose, on the other hand, that storage costs in city j are very high, while

it is very low in city k. In this case, we could potentially find a lower total cost in some scenarios by shipping any excess oil in city j to city k, and thereby incurring lower storage costs.

It should be noted, of course, that the network flow problem will be more complex than the recourse problem as in (6), having additional variables in the form of the quantities to ship between cities.

2.5. **Vector Autoregressions and Scenario Generation.** In many situations, it can be difficult to obtain the joint distribution of the parameters of the problem, particularly because in many cases, the parameters are not independent. An example of dependent parameters might be the demand levels we have discussed for the Oil Problem - since high demand for oil in one city is often associated with high levels of economic activity in that city, it is likely that this will affect the level of economic activity, and therefore the demand for oil, in other cities.

This problem of finding the distribution of the parameters becomes even more complex when we consider the multistage recourse problems, since, for example, demand levels in one city in one stage will likely depend not only on demand levels of the other cities at that stage, but also on demand levels in all cities in previous stages. In other words, the elements of $\vec{b}_t$ are correlated not only with each other, but also with the elements of $\vec{b}_{t-i}$, with $i = 1, 2, \ldots$.

A statistical method for modeling the relationship of $\vec{b}_t$ and previous values of $\vec{b}$ is through a Vector Autoregression (VAR) of order P, which is of the form

$$\vec{b}_t = R_1\vec{b}_{t-1} + R_2\vec{b}_{t-2} + \cdots + R_P\vec{b}_{t-P} + \vec{\mu}_t,$$

where $\vec{b}_t \in \mathbb{R}^n$ for $t = 1, 2, \ldots$, and $R_1, \ldots, R_P \in \mathbb{R}^{n\times n}$. We make the assumption that the error terms in $\vec{\mu}_t$ all have mean 0, may be correlated at time t (which captures the contemporaneous effects of one city's demand on another), but are uncorrelated across time.

Using, for example, T periods of historical data, it is possible to first estimate, using a number of possible criteria, the appropriate order P for the VAR model, and then to find estimates of the coefficient matrices $R_1, \ldots, R_P$ using the method of least squares for each of the n individual linear regressions, to obtain the estimates $\hat{R}_1, \ldots, \hat{R}_P$, as well as estimates of the errors $\hat{\vec{\mu}}_t$.

Now, based on the assumption that $R_1, \ldots, R_P$ will remain unchanged in the future, and that the errors $\vec{\mu}_t$ are uncorrelated across time, it is possible to make forecasts for future values of $\vec{b}$. Supposing that we estimated the coefficients for periods $t = 1, \ldots, T$, where we can think of T as the present period, we could, for example, generate a forecast for $\vec{b}_{T+1}$

by drawing randomly from our set of estimated errors $\hat{\vec{\mu}}_t$ and computing

$$\vec{b}_{T+1} = R_1\vec{b}_T + R_2\vec{b}_{T-1} + \cdots + R_P\vec{b}_{T+1-P} + \hat{\vec{\mu}}_t,$$

since, by assumption, the errors are uncorrelated across time, and, provided we have specified the number of lags P appropriately, will give us an idea of the joint distribution of the errors. In this manner, drawing randomly and with replacement from the set of estimated $\hat{\vec{\mu}}_t$ vectors, we can generate a forecast for several periods into the future, with the sequences of estimated forecasts $\vec{b}_{T+1}, \ldots, \vec{b}_{T+n}$ comprising a single scenario for an n stage recourse problem.

In order to obtain a more accurate picture of the possible evolution of the process, we could compute a large number of forecasts, each time drawing randomly and with replacement from the set of estimated errors. As we compute a large number of forecasts, we can, under our assumptions, obtain a better idea of the possible evolution of the process, with each of the forecasts corresponding to a single scenario to consider in the stochastic linear program. For a more detailed description of these "time series" methods, see [3].

It should be noted that there is a difficult tradeoff to make when using this approach: on the one hand, having a large number of forecasts gives us a more accurate idea of the likelihood of each scenario occurring, yet on the other hand, generating a large number of forecasts, or generating forecasts for more periods into the future, can result in a massive increase in the complexity of the stochastic linear program. Having, for example, N forecasts, looking T periods into the future, would lead to a program with up to $NT + 1$ *sets* of variables for the problem, which could prove to be impractical to solve, owing to the computational time required.

3. Probabilistic Constraints

In the recourse approach we were required to meet demand - or more generally, satisfy all constraints - regardless of the scenario that was realized. However, we may be willing to fail to satisfy some of our constraints a given proportion of the time. Then, we might approach a stochastic linear programming problem such as the Oil Problem described, using probabilistic constraints rather than the recourse approach.

Suppose that some of our constraints *must* be satisfied in any scenario - a good example in the Oil Problem might be the supply constraints. We can include these constraints in the linear program as we did in (2), so these remain binding. On the other hand, we might be willing to have a demand shortfall perhaps 5% of the time in each city, and would satisfy demand in the remaining 95% of scenarios.

In this situation, we could create binary indicator variables $t_{j,s}$ for each city j and scenario s such that

$$t_{j,s} = \begin{cases} 1 & \text{if demand is satisfied in city } j \text{ in scenario } s \\ 0 & \text{otherwise.} \end{cases}$$

Then, for each city j, we would want

$$\sum_{s=1}^{S} t_{j,s} \geq 0.95(S).$$

Subject to these constraints, we would want to maximize the standard objective function $\vec{c}^T\vec{x}$, the total shipping cost.

There are, of course, refinements we might make to the approach described. One possibility might be to ensure that not more than N cities face shortfalls concurrently in any scenario. One way to implement such a constraint would be to require that

$$\sum_{j=1}^{C} t_{j,s} \geq N$$

for each scenario $s = 1, 2, \ldots, S$. In addition, it is possible to set different "tolerance levels" α_j for each city, with

$$\sum_{s=1}^{S} t_{j,s} \geq \alpha_j(S).$$

Thus, we might require, for example, that we meet demand at least 99% of the time in one city, and only 80% in another.

This approach could be, in many applications, more appealing than the recourse approach, particularly if we are in reality willing to violate some of the constraints a small percentage of the time. However, we should bear in mind that these constraints do not take into account the *extent* of the failure to meet any constraint. In the context of the Oil Problem, we would be indifferent between a small shortfall in demand in some scenario, and an extremely large shortfall. The recourse approach, in contrast, *does* account for the extent of shortfalls, effectively through the recourse costs, since a large shortfall would require us to ship a greater recourse quantity at a higher cost. In this case, we can think of the recourse costs $\vec{r}_s$ as being the penalty per unit shortfall in demand, rather than the recourse cost of shipping. In addition to this, using probabilistic constraints will mean that we now must solve an Integer Linear Program, which, when combined with a large number of scenarios to consider, can become computationally difficult to solve.

4. Conclusion

We have shown a number of the more common approaches to dealing with linear programming problems where some or all of the parameters are random. The common theme of the methods described are that they seek to convert a problem with random parameters, which we may not know how to solve, into a *similar* problem with known parameters, which we *do* know how to solve. It should be borne in mind, however, that these related problems can be highly complex in terms of the number of variables and constraints, and consequently can be computationally more difficult and time-consuming to solve.

5. Appendix

5.1. **Extensive form for the Linear Program described in (6):** The extensive form of the problem we described in (6) is shown here.

Minimize

$$\begin{pmatrix} 20 \\ 20 \\ 10 \\ 50 \\ 10 \\ 15 \\ 40*2/3 \\ 40*2/3 \\ 20*2/3 \\ 100*2/3 \\ 20*2/3 \\ 30*2/3 \\ 60*1/3 \\ 60*1/3 \\ 30*1/3 \\ 150*1/3 \\ 30*1/3 \\ 45*1/3 \end{pmatrix}^T \begin{pmatrix} x_{1,1} \\ x_{1,2} \\ x_{1,3} \\ x_{2,1} \\ x_{2,2} \\ x_{2,3} \\ y_{1,1,1} \\ y_{1,2,1} \\ y_{1,3,1} \\ y_{2,1,1} \\ y_{2,2,1} \\ , y_{2,3,1} \\ y_{1,1,2} \\ y_{1,2,2} \\ y_{1,3,2} \\ y_{2,1,2} \\ y_{2,2,2} \\ y_{2,3,2} \end{pmatrix}$$

subject to

$$\left(\begin{array}{cccccc|cccccc|cccccc}
-1 & -1 & -1 & 0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
-1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1
\end{array}\right)
\left(\begin{array}{c}
x_{1,1} \\ x_{1,2} \\ x_{1,3} \\ x_{2,1} \\ x_{2,2} \\ x_{2,3} \\ y_{1,1,1} \\ y_{1,2,1} \\ y_{1,3,1} \\ y_{2,1,1} \\ y_{2,2,1} \\ , y_{2,3,1} \\ y_{1,1,2} \\ y_{1,2,2} \\ y_{1,3,2} \\ y_{2,1,2} \\ y_{2,2,2} \\ y_{2,3,2}
\end{array}\right)
\leqslant
\left(\begin{array}{c}
-300 \\ -200 \\ 300 \\ 100 \\ 100 \\ -300 \\ -200 \\ 200 \\ 250 \\ 50
\end{array}\right).$$

Note that the first 5 constraints correspond to the first scenario, and the second 5 correspond to the second scenario.

5.2. **Mathematica code:**

```
M = {{-1, -1, -1, 0, 0, 0},
  {0, 0, 0, -1, -1, -1},
  {1, 0, 0, 1, 0, 0},
  {0, 1, 0, 0, 1, 0},
  {0, 0, 1, 0, 0, 1}
  }

  b1 = {-300, -200, 300, 100, 100}
  b2 = {-300, -200, 200, 250, 50}
  bE = (2/3)*b1 + (1/3)*b2
  c = {20, 20, 10, 50, 10, 15}

  rE = LinearProgramming[c, M, bE]
  M.rE

  bigM = {
  {-1, -1, -1, 0, 0, 0, -1, -1, -1, 0, 0, 0,0, 0, 0, 0, 0, 0},
  {0, 0, 0, -1, -1, -1, 0, 0, 0, -1, -1, -1, 0, 0, 0, 0, 0, 0},
  {1, 0, 0, 1, 0, 0,  1, 0, 0, 1, 0, 0,  0, 0, 0, 0, 0, 0},
  {0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0,0, 0, 0, 0, 0, 0},
  {0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0},
  {-1, -1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, -1, -1, 0, 0, 0},
  {0, 0, 0, -1, -1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, -1, -1},
  {1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0},
  {0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0},
  {0, 0, 1, 0, 0, 1,0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1}
  }

  B = {-300, -200, 300, 100, 100, -300, -200, 200, 250, 50}
  shippingcost = {20, 20, 10, 50, 10, 15,
                  40*2/3, 40*2/3, 20*2/3, 100*2/3, 20*2/3, 30*2/3,
                  60*1/3, 60*1/3, 30*1/3, 150*1/3, 30*1/3, 45*1/3}

  result2 = LinearProgramming[shippingcost, bigM, B]

  This gives us the optimal expected total cost:
  shippingcost.result2

  Now compute the costs we would have incurred had we known demand beforehand:
  r1 = LinearProgramming[{20, 20, 10, 50, 10, 15},
  M, {-300, -200, 300, 100, 100}]
```

```
{20, 20, 10, 50, 10, 15}.r1

r2 = LinearProgramming[{20, 20, 10, 50, 10, 15},
M, {-300, -200, 200, 250, 50}]

{20, 20, 10, 50, 10, 15}.r2

2/3*8500 + 1/3*7500
```

Modify the supplies so that total capacity exceeds total demand,
and we could potentially oversupply:

```
r = LinearProgramming[
{20, 20, 10, 50, 10, 15,
 200*2/3, 200*2/3, 100*2/3, 500*2/3, 100*2/3, 150*2/3,
 200*1/3, 200*1/3, 100*1/3, 500*1/3, 100*1/3, 150*1/3},
bigM,
{-350, -220, 300, 100, 100, -350, -220, 200, 250, 50}
]
```

city 2 gets an oversupply in s1, city 1 gets an oversupply in s2:

```
bigM.r
```

Now create very high storage costs:

```
storagecosthigh = {0, 0, 4000*2/3, 30000*2/3, 20000*2/3, 0, 0,
50000*1/3, 20000*1/3, 50*1/3}

shippingcosthigh = {20, 20, 10, 50, 10, 15,
200*2/3, 200*2/3, 100*2/3, 500*2/3, 100*2/3, 150*2/3,
200*1/3, 200*1/3, 100*1/3, 500*1/3, 100*1/3, 150*1/3}

totalexpectedcost2 = shippingcosthigh + storagecosthigh.bigM

r2 = LinearProgramming[totalexpectedcost2,
bigM, {-350, -220, 300, 100, 100, -350, -220, 200, 250, 50}]
```

We see that the extent of the possible oversupply is a bit less in the cities
where we put in the absurdly high storage costs:

```
bigM.r2
```

```
In this case, we're dropping the constant in the objective function (storage
costs are incurred only for amounts in excess of demand), so actually
interpreting something like

totalexpectedcost2.r2

will give an overestimate. This was just to illustrate how storage costs can change
the optimal values.
```

References

[1] Suvrajeet Sen and Julia L. Higle (1999): *An Introductory Tutorial on Stochastic Linear Programming Models,* INTERFACES 29: 2 March–April 1999 (pp. 33–61).
[2] Gerard Cornuejols and Reha Tütüncü (2005): *Optimization Methods in Finance.*
[3] Robert H. Shumway and David S. Stoffer (2010): *"Time Series Analysis and Its Applications."*