Paper T11-2009

Decision Making Using PROC OPTMODEL

Joseph Czyzyk, Central Michigan University Research Corporation,
Mount Pleasant, MI

ABSTRACT

How do you plan for an uncertain future? A common approach for tackling this problem is to generate a set of possible outcomes for the future, commonly called scenarios. Each scenario has some probability of occurrence, and the decision-maker is responsible to use the scenarios to make a wise decision in the current period to be ready for the future. If the decision problem can be formulated as a linear program, the uncertainty can be captured in the model using a stochastic programming formulation.

We will present how to use PROC OPTMODEL to write the linear programming formulation of a planning problem. We will show how scenarios can be written to capture the uncertainty in future periods of the planning horizon. The stochastic program will be formulated and solved using PROC OPTMODEL. We will show how optimizing over all of the future scenarios together at one time provides a better solution than optimizing over each scenario individually.

We will see that the stochastic programming approach optimizes over all of the scenarios at once by minimizing the expected cost over all the scenarios. The stochastic solution is, in some sense, the best "hedge" against future uncertainties.

INTRODUCTION

Many business problems such as scheduling problems, assignment problems, and product mix problems can be formulated as linear programs. The solution of these problems chooses optimal values of decision variables according to a given objective function. The business can then implement the plan to obtain the best business result.

In this paper we will look at a simple planning model with one product over two time periods. The problem can be easily expanded to cover a number of products or expanded to a number of time periods. The problem is artificially simple since realistic business problems involve thousands of variables and constraints. Even so, salient points of making decisions are shown. We will look at how PROC OPTMODEL implements linear programs.

The problem will be extended to consider situations where the problem data are not exactly defined. Our example problem requires an estimate of future demand, and we will see that different estimates can lead to very different solutions. Concepts from stochastic linear programming will be used to determine the best solution in these situations.

LINEAR PROGRAMMING

Linear programs are decision-making models. Unlike regression or neural networks where the relationships between variables in the data are unknown, linear programs assume that the relationships between variables are clearly defined and that there is a measure which can be computed to say that one solution is better than another. For example, if a business can choose to produce products A, B or C all from the same raw materials, it would want to produce the right mix of A, B and C to maximize profit. We assume that there are equations which define what is involved to produce each product from raw materials. We also assume that the profit margin for each product is known. The solution to the problem is the value of each decision variable.

Linear programs contain parameters, decision variables, constraints and an objective function. We will explain these in the context of the problem that we will analyze in this paper. The example we will consider involves when to purchase and store a product for both current and future use. We'll state the problem as natural gas planning, but it can easily be translated to water reservoir planning or any other type of inventory management with fluctuating costs and demand. A company buys, stores and distributes natural gas. It must meet the demand of its customers and can do so by purchasing gas or taking it from its storage reserves. There is a cost for putting gas into storage so gas is only stored when the price for a future time period is expected to be sufficiently high to warrant the additional storage expense.

COMPONENTS OF LINEAR PROGRAMS

Decision variables are the controllable inputs to the model. The decision maker can choose the values of the decision variables usually with some restrictions imposed by constraints in the model. The solution of the LP will give optimal values of the decision variables. In the example, the amount of gas to buy and the amount to use from storage in each time period will be the decision variables.

Parameters are additional input values to the model, but they are not under the control of the decision maker. Example parameters will be the purchase price of the gas, the storage cost and the outside demand. The parameters in some sense define the problem to be optimized.

Constraints are equations or inequalities which restrict the values of the decision variables. For example, it is not possible to store or buy a negative amount of gas. The value of gas inventory is equal to the previous inventory minus the amount taken from inventory. All of the restrictions on the decision variables are formulated as constraints.

The **objective function** is a function of the decision variables to optimize (maximize or minimize). In the example the objective is to minimize total cost of gas purchases and storage. In other problems, it could be to maximize profit, minimize delay, minimize error, etc.

Since we are considering linear programs, all constraints and the objective function must be linear functions of the decision variables. It is possible to include nonlinear functions, but the problem would be considered a nonlinear program and would require different solution methods. PROC OPTMODEL allows users to formulate both linear and nonlinear models and contains solutions methods for both.

EXAMPLE VERBAL PROBLEM FORMULATION

We will consider the purchasing and storage decision of natural gas for a 2-year time period. We assume that the demand for the first year is known to be 10 units at a cost of $1 per unit. Gas can be purchased and stored, and the cost to store a unit of gas from one year to the next is $0.10. The demand for year 2 is expected to be 11 units with no increase in cost. What is the optimal purchase/store solution?

The **parameters** of the model are as follows:

- Periods: year 1, year 2
- Demand for each year
- Purchase price for each year
- Inventory cost for each year

In table form,

Period	**Price**	**Demand**	**Inventory Cost**
1	1.0	10	0.10
2	1.0	11	0.10

The **decision variables** are:

- Amount of gas to buy each year
- Amount of gas to use from inventory each year
- Amount of gas in inventory each year

The **constraints** are:

1. Meet Demand: The amount of gas bought and taken from storage must meet demand.
2. Inventory Balance: The gas inventory equals the inventory in the previous year plus the amount of gas purchased minus the gas demand.

The **objective function** is to minimize the total cost of gas purchase and gas storage.

Note that the problem could be formulated differently with variables for the amount of gas put into storage, for example. The constraints would change appropriately. In addition there could be limits on the size of the storage facility, amount purchased, etc.

EXAMPLE PROBLEM FORMULATION IN PROC OPTMODEL

Although it is possible to formulate the problem in terms of a number of variables *x* with different subscripts, SAS provides a convenient modeling language within PROC OPTMODEL for formulating problems. Please see the document *SAS/OR 9.1.3 User's Guide: Mathematical Programming 3.2* for more information.

We start the problem statement with PROC OPTMODEL.

```
proc optmodel;

/* Declare parameters of the model */

set<num> Periods;

number Price{Periods};
number Demand{Periods};
number InventoryCost{Periods};
```

The *set* statement indicates that there will be a group of numbers (*num*) called *Periods*. In the problem formulation we will index our variables and parameters using the values of this set.

We also declare three parameters which will be numbers (*num*) with each variable indexed over the set *Periods*. That is, there will be a variable *Price* for each of the periods in the model and similarly for the other variables.

```
/* Assign parameters */

Periods = 1..2;

Price[1] = 1.0;  Demand[1] = 10;     InventoryCost[1] = 0.1;

Price[2] = 1.0;  Demand[2] = 11;     InventoryCost[2] = 0.1;
```

The statements above assign values to the parameters that have already been declared.

```
/* Declare all decision variables in the model */

var Buy{Periods} >= 0;
var UseFromInventory{Periods} >= 0;
var Inventory{Periods} >= 0;
```

Here we declare the decision variables in the model. Once the problem is formulated and solved, the values of these variables will be the policy to be implemented by the gas company. Note that there are decision variables for each time period and that we restrict all of them to be greater than or equal to zero.

```
/* Specify the objective function */

min TotalCost = sum {t in Periods}
                (Price[t] * Buy[t] + InventoryCost[t] * Inventory[t]);
```

The objective function is straightforward. We want to minimize the total cost of buying and storing gas. Notice that this statement works for problems with as many periods as we have in the model. PROC OPTMODEL will expand the equation regardless of the number of periods.

```
/* Specify the constraints */

constraint MeetDemand{t in Periods}:
    Buy[t] + UseFromInventory[t] >= Demand[t];
```

```
constraint InventoryBalance1{t in Periods: t=1}:
    Inventory[t] = Buy[t] - Demand[t];

constraint InventoryBalance{t in Periods: t>1}:
    Inventory[t] = Inventory[t-1] + Buy[t] - Demand[t];
```

The constraints relate the parameters and decision variables together. The first constraint states that the amount of gas purchased and used from inventory must be greater than or equal to the demand for every time period.

The last two constraints are actually one constraint but broken down into two cases. The second constraint applies only when the period (here denoted *t*) is 1. [We have assumed that the input dataset will have periods starting at 1 and increasing by 1.] The amount of gas in inventory will equal the amount bought less the gas used to meet the demand. The third constraint is for all periods except the first and it shows the inventory balance equation, namely, the inventory in a time period is equal to the previous inventory plus the amount of gas purchased in excess of demand.

```
expand;
solve;
print Price Demand;
print Buy UseFromInventory Inventory;
print TotalCost;
quit;
```

The *expand* statement will print out the final problem formulation showing all of the parameters, decision variables, constraints and the objective function. This is useful for debugging small problem formulations. I would recommend using artificially small datasets as input and using the *expand* function. Once you are convinced that the model formulation is correct, you can run problems.

The *solve* statement invokes an appropriate solver. Since linear programs can be unbounded, infeasible or solvable, it is important to check the solution status.

The *print* statements output the problem parameters, decision variables and objective function.

CASE STUDY

THREE SCENARIOS

The gas company would like to analyze the optimal solutions for three different scenarios for the weather in time period 2. The details on the scenarios are as follows:

Period	**Price**	**Demand**	**Inventory Cost**	**Probability**	**Scenario**
1	1.0	10	0.10		
2	1.5	16	0.10	0.5	Harsh winter
2	1.0	8	0.10	0.2	Mild winter
2	1.0	11	0.10	0.3	Regular winter

The problem was formulated for each of the three scenarios by changing the values of the parameters with the following results:

Scenario	**Buy in Period 1**	**Use from inventory in period 1**	**Inventory in period 1**	**Buy [2]**	**Use from inv. [2]**	**Inventory [2]**	**Total Cost**
Harsh	26	0	16	0	16	0	27.6
Mild	10	0	0	8	0	0	18
Regular	10	0	0	11	0	0	21

OPTIMIZE THE AVERAGE SCENARIO

The company was surprised to see that the optimal solution varied so much between the scenarios. The company made the decision to compute the weighted average of the three scenarios and then execute the optimal solution for that problem. Please note that this is not a good practice but is done here for illustrative purposes.

The problem becomes:

Period	Price	Demand	Inventory Cost
1	1.0	10	0.10
2 avg	1.25	12.9	0.10

with the solution:

Buy in Period 1	Use from inventory in period 1	Inventory in period 1	Buy [2]	Use from inv. [2]	Inventory [2]	Total Cost
22.9	0	12.9	0	12.9	0	24.19

Let's evaluate this "average optimal" solution under each of the three scenarios (using the demand for each scenario). In the first period we buy 22.9 units of gas at $1 per unit. In the regular scenario, we meet 11 units of demand from inventory and keep the remaining 1.9 units in inventory. In the harsh scenario, we take the 12.9 units from inventory and buy 3.1 units to meet the demand of 16. In the mild scenario, we meet the demand of 8 from inventory and let 4.9 units remain in inventory. The total cost is computed using the formula for the objective function.

Scenario	Buy in Period 1	Use from inventory in period 1	Inventory in period 1	Buy [2]	Use from inv. [2]	Inventory [2]	Total Cost
Harsh	22.9	0	12.9	3.1	12.9	0	28.84
Mild	22.9	0	12.9	0	8	4.9	24.68
Regular	22.9	0	12.9	0	11	1.9	24.38

The weighted average of the total cost over all the scenarios is $26.67 greater than $24.19, the total cost of the average optimal solution.

STOCHASTIC PROGRAMMING

Stochastic linear programming is a modeling framework for handling uncertainty in some of the problem data. As in the example above, the uncertainty can be expressed as scenarios allowing the problem to be solved taking the uncertainty into account rather than finding a way to deal with it afterwards. Volumes have been written on stochastic optimization, but here we want to demonstrate how the linear programming problem can be formulated as a stochastic program. PROC OPTMODEL can expand our formulation and solve the problem.

In the original problem formulation we solved the problem separately for each scenario. As a result, the solution provided different values for the amount of gas to purchase in the first period. This number varied depending on the price and demand for gas in the future period. Since the future is unknown and a decision must be made in the current period, the values of all first period variables must be the same for all scenarios. This constraint on the variables is called a non-anticipativity constraint. We will incorporate this into our model by only have one set of decision variables for the first period. In contrast, there will be different decision variables (buy, UseFromInventory, Inventory) for the second period for each scenario. We will formulate the problem to minimize the total cost of the first-period decisions plus the *expected* total cost of the second-period decisions weighted by the probability of their occurrence. We will create separate second-period decision variables for each scenario and constraints as well.

STOCHASTIC PROBLEM FORMULATION

```
/* This program represents the stochastic programming version of the natural gas
   planning problem.
 */

proc optmodel;

/* Declare parameters of the model */

set<num> Periods;
set<string> Scenarios{Periods};

number Price{t in Periods, Scenarios[t]};
number Demand{t in Periods, Scenarios[t]};
number InventoryCost{t in Periods, Scenarios[t]};
number Probability{t in Periods, Scenarios[t]};
string ParentScenario{t in Periods, Scenarios[t]};
```

Note the new set *Scenarios* which represents all of the possibilities in a given time period. The first time period must have one scenario but the remaining time periods may have numerous scenarios. We include the parameter *Probability* to indicate the likelihood of a given scenario in a time period. Also there is a parameter *ParentScenario* to indicate which scenario in the previous time period is the parent. This allows for a complex scenario tree representation with different numbers of scenarios in future time periods. In this example all scenarios are in the second time period and arise from the base scenario in time period 1. Note that all parameters will now have two subscripts to show the dependence on the period and the scenario.

```
/* Assign parameters */

Periods = 1..2;
Scenarios[1] = {'Regular'};
Scenarios[2] = {'Regular', 'Harsh', 'Mild'};

Price[1,'Regular'] = 1.0;            Demand[1,'Regular'] = 10;
InventoryCost[1,'Regular'] = 0.1;    Probability[1,'Regular'] = 1.0;

Price[2,'Regular'] = 1.0;            Demand[2, 'Regular'] = 11;
InventoryCost[2, 'Regular'] = 0.10;  Probability[2, 'Regular'] = 0.3;

Price[2,'Harsh'] = 1.5;              Demand[2,'Harsh'] = 16;
InventoryCost[2,'Harsh'] = 0.10;     Probability[2,'Harsh'] = 0.5;

Price[2,'Mild'] = 1.0;               Demand[2,'Mild'] = 8;
InventoryCost[2,'Mild'] = 0.10;      Probability[2,'Mild'] = 0.2;

ParentScenario[2,'Regular'] = 'Regular';
ParentScenario[2,'Harsh'] = 'Regular';
ParentScenario[2,'Mild'] = 'Regular';

/* Declare all decision variables in the model */

var Buy{t in Periods, Scenarios[t]} >= 0;
var UseFromInventory{t in Periods, Scenarios[t]} >= 0;
var Inventory{t in Periods, Scenarios[t]} >= 0;
```

Note that all variables also have both period and scenario subscripts.

```
/* Specify the objective function */

min TotalCost = sum {t in Periods, scen in Scenarios[t]}
                (Price[t, scen] * Buy[t, scen]
              + InventoryCost[t, scen] * Inventory[t, scen]) * Probability[t, scen];
```

The objective function has changed to show that the terms are weighted by the probability of the scenarios.

```
/* Specify the constraints */

constraint MeetDemand{t in Periods, scen in Scenarios[t]}:
    Buy[t, scen] + UseFromInventory[t, scen] >= Demand[t, scen];

constraint InventoryBalance1{t in Periods, scen in Scenarios[t]: t=1}:
    Inventory[t, scen] = Buy[t, scen] - Demand[t, scen];

constraint InventoryBalance{t in Periods, scen in Scenarios[t]: t>1}:
    Inventory[t, scen] =
    Inventory[t-1, ParentScenario[t,scen]] + Buy[t, scen] - Demand[t, scen];
```

The constraints have also changed to incorporate the different scenarios. Note that constraints which involve the previous time period reference the parent scenario.

```
expand;
solve;
print Price Demand;
print Buy UseFromInventory Inventory;
print TotalCost;
quit;
```

CASE STUDY EXAMPLE USING STOCHASTIC PROGRAMMING

If we consider the same problem in our case study, we obtain the following solution:

Scenario	**Buy in Period 1**	**Use from inventory in period 1**	**Inventory in period 1**	**Buy [2]**	**Use from inv. [2]**	**Inventory [2]**	**Total Cost**
Harsh	18	0	8	8	8	0	30.8
Mild	18	0	8	0	8	0	18.8
Regular	18	0	8	3	8	0	21.8

The expected total cost of the solution (weighted by the probability of the scenarios) is $25.7. Notice that this is lower than $26.67, the expected total cost of using the “average” scenario solution. The difference between $26.67 and $25.7 or $0.97 is the value of the stochastic solution.

Notice that executing the stochastic solution puts the company in the best position to handle any of the three scenarios which might occur in the second period. If the future were known, then the company could optimize its solution separately for any of the scenarios as we did in a previous section. The difference between the total cost for those solutions and the stochastic solution is known as the value of perfect information.

CONCLUSION

PROC OPTMODEL provides a powerful modeling language for both linear and nonlinear programs. Problems can be formulated in a succinct manner and read data from SAS datasets. The procedure also includes solvers to compute solutions to the formulated problems.

In this example, we showed how to formulate a simple linear program and obtain a solution. We have also shown that the common practice of planning for the average future scenario may not be the best solution. Stochastic programming formulations are larger and more difficult to solve but provide more valuable solutions for planning.

REFERENCES

SAS/OR 9.1.3 User's Guide: Mathematical Programming 3.2, Volumes 1-4.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name: Joseph Czyzyk
Enterprise: Central Michigan University Research Corporation
Address: 2625 Denison Drive
City, State ZIP: Mount Pleasant, MI 48858
Work Phone: 989-774-1622
E-mail: Joe.Czyzyk@cmich.edu