**Python Examples:**

PuLP Package:

First two examples (Inventory Question and Quiz Question).

Note, the PuLP Package can call Gurobi, but we are going to use the built-in Solver today.

More details: <https://pypi.org/project/PuLP/>

*Note, I like using the PuLP package (initially) because it looks a lot like how the formulation looks when you write it line-by-line on the board (or type it).*

**Question 1**

A company wants to plan production for the ensuing year to minimize the combined cost of production and inventory costs. In each quarter of the year, demand is anticipated to be 130, 160, 250, and 150 units, respectively. The plant can produce a maximum of 200 units each quarter. The product can be manufactured at a cost of $15 per unit during the first quarter, however the manufacturing cost is expected to rise by $1 per quarter. Excess production can be stored from one quarter to the next at a cost of $1.50 per unit, but the storage facility can hold a maximum of 60 units. How should the production be scheduled so as to minimize the total costs? Formulate an LP model.

Background:

Production Inventory Problem, sometimes referenced as Inventory Smoothing Problem

Flow Conservation: What Goes In = What Comes Out

Production = Pi this will signify production in time period i

Storage = Si this will signify ending storage (inventory) time period i (i.e., the amount of storage at the end of period i)

Picture:

P4

S4

150

Period 4

P2

P1

S2

Period 2

S1

Period 1

S0

…

160

130

Formulation:

Indexed Set:

i = (1, 2, 3, 4); time period

Decision Variables:

Pi = production in period i

Si = inventory (ending) of period i

*\*Note, since problem does not specify a “beginning inventory” nor “ending inventory” constraint; we will assume S0 and S4 are 0.*

Objective:

Minimize Total Costs = 15P1 + 16P2 + 17P3 + 18P4 + 1.5S1 + 1.5S2 + 1.5S3

Constraints:

Pi <= 200 for all i (Plants can only produce up to 200 units); upper bound

Si <= 60 for all i (Storage can only hold up to 60 units); upper bound

P1 = S1 + 130 (Period 1 equilibrium)

S1 + P2 = S2 + 160 (Period 2 equilibrium)

S2 + P3 = S3 + 250 (Period 3 equilibrium)

S3 + P4 = 150 (Period 4 equilibrium)

Si >= 0 (Non-negativity)

Pi >= 0 (Non-negativity)

**Question 2: (Quiz Version A)**

ABC Gas Company blends gasoline and then sells it to customers. Three different grades of gasoline are used to fill customer orders. Next month there will be two customers. ABC Gas Company wishes to minimize its cost of fulfilling these two orders by determining the number of gallons of gasoline for each grade to use for each specific customer. Formulate as an LP model (i.e., clearly define decision variables, objective, and constraints).

The three grades of gas are presented below, with their associated specifications.



The customer orders are presented below, with their associated specifications.



You can assume a linear blend (i.e., no chemical reactions).

For example, 1 gallon of 87 octane and 1 gallon of 89 octane will yield 2 gallons of 88 octane.

Solution:

Decision Variables:

xij: the number of gallons of gasoline grade i (i=1,2,3) sold to customer j (j=1,2)

Objective: Minimize Cost  


Subject To:

Supply Constraints



Demand Constraints

 (Note: the demand constraints could be = instead of ≥)

Customer Blend Constraints



Non-negativity Constraints



**Gurobi Examples:**

Note, if you installed Gurobi on your local machine, then there are a number of Python examples (and data files) already saved to your machine. On Wilck’s Machine they are saved locally at:

**C:\gurobi912\win64\examples**

*With the data folder having data files and the python folder having python files. If you move the python files to a new folder (or directory), then you may need to change where they are pointing to for access to the data file.*

Code Examples:

<https://www.gurobi.com/resource/functional-code-examples/>

Model Separation:

<https://www.gurobi.com/documentation/9.1/examples/model_data_separation_in_p.html>

Diet:

<https://www.gurobi.com/documentation/9.1/examples/diet4_py.html>

Gurobi has a lot of options. You may need to review dictionaries and lists with respect to your python repertoire. Unfortunately, it will not always look like how you would write it by-hand.

We will focus on the diet.py first.

Formulation:

i = nutrition categories (i = 1, …, I)

j = food types (j = 1, …, J)

Data:

Ui = upper limit on nutrition needs per i

Li = lower limit on nutrition needs per i

Cj = cost of food j

Aij = nutrition i amount of food j

Variables:

Xj = amount of food j purchased

Objective:

Minimize Cost = (Min Cost)

Constraints:

(Lower and Upper Limit)

(Non-negativity)