Operational Research | ECE AUTh | 8th semester

Kitsios Konstantinos 9182

May 5, 2020

Problem

The purpose of this project is to use Operational Research techniques in order to minimize the production cost of a motor factory. The company produces two types of components, let's say **type A** and **type B**. The factory assemply line can only handle one type per week and the decision to switch types can only be taken an the end of each week with an extra cost of **500$**. The maximum production per week is **100** for type A and **80** for type B. Last week the factory closed with assemply line being at type A with a stock of 125 type A components and 143 type B components. Furthermore, the annual storage cost is calculated with an interest rate of 19.5% per year. The cost per type A component is **225$** and per type B component is **310$**. The stock policy of the company states that at the end of each week the stock is at least the 80% of the demand of next week. The demand for the next 9 weeks is shown at the table (below | attached).

Modeling

Constraints

First, let's denote with

$$x_i\,, i = 1, 2, ..., 18$$

the production of the two types for the 9 weeks and with

$$y_i\,, i = 1, ..., 18$$

the corresponding (given) demand. Obviously,

$$x_i \geq 0,\ i = 1, ..., 18 \tag{1}$$

We make the assumption that the first 9 elements of $\boldsymbol{x}$ are for type A and the last 9 elements are for type B.

We will start with the constraint that only one of x_i and x_{i+9} for $i = 1, ..., 9$ can be nonzero each time. Since we can only use linear operations, the product

of the two variables cannot be used. Instead we could model this condition with the following:

$$x_i + x_{i+9} = max\{x_i, x_{i+9}\},\ i = 1, ..., 9$$

This ensures that either $x_i = 0$ or $x_{i+9} = 0$. The $max\{\}$ operator will be implemented in linear terms using the $big - M$ notation. Thus, if we denote

$$m_i = max\{x_i, x_{i+9}\},\ i = 1, ..., 9$$

we have the following constraints for $i = 1, ..., 9$:

$$m_i \geq x_i \tag{2}$$

$$m_i \geq x_{i+9} \tag{3}$$

$$m_i \leq x_i + Mb_i \tag{4}$$

$$m_i \leq x_{i+9} + M(1 - b_i) \tag{5}$$

where

$$b_i \epsilon \{0, 1\}$$

is a binary variable: $b_i = 0 \Longrightarrow$assemply line is set to type A in week i and $b_i = 1 \Longrightarrow$assemply line is set to type B in week i, and M is a big enough constant number. Equations $(2) - (5)$ ensure that $m_i = max\{x_i, x_{i+9}\}$ and the equation below assigns the maximum to be equal to their sum:

$$m_i = x_i + x_{i+9},\ i = 1, ..., 9 \tag{6}$$

Now the equations $(2) - (6)$ state **in linear terms** that **either** $x_i = 0$ **or** $x_{i+9} = 0$, $i = 1, ..., 9$

Next we will take care of the maximum production rate:

$$x_i \leq 100,\ i = 1, ..., 9 \tag{7}$$

$$x_i \leq 80,\ i = 10, ..., 18 \tag{8}$$

and the 80% stock supply for the next week:

$$\sum_{k=1}^{i}(x_k - y_k) + 125 \geq 0.8x_{i+1},\ i = 1, ..., 8 \tag{9}$$

$$\sum_{k=1}^{i}(x_k - y_k) + 143 \geq 0.8x_{i+1},\ i = 10, ..., 17 \tag{10}$$

Finally, we will need a binary variable to state wether we had a **switch** of the assembly line each week. We already have b_i to decide the stattus of the assemply line each week($b_i = 0 \rightarrow typeA$, $b_i = 1 \rightarrow typeB$) so defining c_i is as simple as taking the XOR of $b_i$for two subsequent days:

$$c_i = b_i XOR b_{i-1}, i = 1, ..., 9$$

where we suppose $b_0 = 0$ because at the last week the assemply line was at type A mode. One way to implement XOR between two binary variables in linear form is the following(easily proved with the truth table):

$$c_i = |b_i - b_{i-1}| = max\left\{b_i - b_{i-1}, -b_i + b_{i-1}\right\}, i = 1, ..., 9$$

and the for the max operator:

$$c_i \geq b_i - b_{i-1} \tag{11}$$

$$c_i \geq -b_i + b_{i-1} \tag{12}$$

Note that we did not use the $big - M$ notation here that would have generated 2 more equations to ensure the max operator. This happens for a reason: c_i will be used inside the minimization objective i.e. the algorithm will try to minimize an increasing function of c_i. This means the algorithm itself will make c_i *as small as possible* but yet satisfying $(11), (12)$ which ensures that c_i is exactly the maximum value and not a value larger than maximum which also satisfies $(11), (12)$

Objective

Now that we have set up all the constraints $(1) - (12)$ we procceed in defining the minimization objective: the total cost J. The cost consists of 3 elements:

1. The production cost J_1
2. The assemply line switch cost J_2
3. The storage interest rate cost J_3

For J_1we have

$$J_1 = \sum_{i=1}^{9} 225x_i + 310x_{i+9}$$

Similarily for J_2

$$J_2 = \sum_{i=1}^{9} c_i * 500$$

and for J_3

$$J_3 = \sum_{i=1}^{9} \left\{ \left[\sum_{k=1}^{i} (x_k - y_k) + 125 \right] 0.84375 + \left[\sum_{k=1}^{i} (x_{9+k} - y_{9+k}) + 143 \right] 1.1625 \right\}$$

where the quantity in square brackets [] is the stock at the end of week i (type A in first brackets and type B in second brackets). This is obtained by simply adding all the production up to that week, substracting all the demand and finally adding the initial stock(125 for type A and 143 for type B). The square bracket coefficients come from the annual interest rate when converted in weekly

interest rate: 0.195/52 and multiplied by the cost of each type(225 for type A and 310 for type B). Hence, our objective is the following

$$min\left\{J_1 + J_2 + J_3\right\} =$$

$$min\left\{\sum_{i=1}^{9} 225x_i + 310x_{i+9} + c_i * 500 + \left[\sum_{k=1}^{i}(x_k - y_k) + 125\right] 0.84375 + \left[\sum_{k=1}^{i}(x_{9+k} - y_{9+k}) + 143\right] 1.1625\right\}$$

Results

For the solution we used the ***Pyomo***[1] library for python. The above procedure yielded 12 linear constraints $(1) - (12)$ and a linear minimization objective and since there are binary variables we have a Mixed Integer Linear Problem(MILP) for which we used the ***GLPK***[2] solver . The full code is in this repository as well as in the same .zip with this file. Below we present the results:

Figure 1:

```
(pyomo) kitsiosk@kitsiosk:~/Desktop/OperationalResearch/pyomo$ python project.py
Print values for all variables
x[1] 10.0
x[2] 45.0
x[3] 100.0
x[4] 0.0
x[5] 0.0
x[6] 0.0
x[7] 0.0
x[8] 0.0
x[9] 0.0
x[10] 0.0
x[11] 0.0
x[12] 0.0
x[13] 2.0
x[14] 48.0
x[15] 47.0
x[16] 46.0
x[17] 31.0
x[18] 0.0
m[1] 10.0
m[2] 45.0
m[3] 100.0
m[4] 2.0
m[5] 48.0
m[6] 47.0
m[7] 46.0
m[8] 31.0
m[9] 0.0
b[1] 0.0
b[2] 0.0
b[3] 0.0
b[4] 1.0
b[5] 1.0
b[6] 1.0
b[7] 1.0
b[8] 1.0
b[9] 1.0
c[1] 0.0
c[2] 0.0
c[3] 0.0
c[4] 1.0
c[5] 0.0
c[6] 0.0
c[7] 0.0
c[8] 0.0
c[9] 0.0
Objective value:
90432.65000000001
(pyomo) kitsiosk@kitsiosk:~/Desktop/OperationalResearch/pyomo$
```

We see an interesting and expected structure in the results: The ***assemply line only switches once*** which is totally expected and an indicator that the code works well. As wee see $c_4 = 1$ and $c_i = 0\, for\, i \neq 4$. We can also confirm this if we take a look at b_i values where $b_i = 0\, for\, i = 1, 2, 3$ and $b_i = 1,\ for\, i = 4, ..., 9$ which is also expected from the modeling. The same arguements hold

for values of $\boldsymbol{x}$ and $\boldsymbol{m}$ as well: $m_i = max\{x_i, x_{i+9}\}, i = 1, ..., 9$ as desired and furthermore $\boldsymbol{x}$ follows the pattern of $\boldsymbol{b}$: It gives non zero production of type A for the first 3 weeks and zero production of type A for weeks 4-9 while for type B the opposite holds. **All the above serve as testimonials for the correctness of code**. Finally, the calculated minimum value for the optimization objective is **90.432, 65$** total cost for the factory.

References

[1] http://www.pyomo.org/

[2] https://www.gnu.org/software/glpk/

[3] Hillier, F. S., & Lieberman, G. J. (1967). Introduction to operations research. San Francisco: Holden-Day.