# Price Recommendation in Inventory Management Optimization Problem

Ahmed Jawad

Wednesday 11th December, 2024

## Context

We are faced with an inventory management and demand forecasting problem for an online Retailer. The objective is to maximize the sum of revenues over a 90-day period by adjusting the price and quantity forecasts. We need to consider constraints related to min stock levels for all times and price variations. Additionally, we need to account for shipments arriving at specific days with known quantities of items.

```
 --------------------------------------
/                                      \
|       Inventory Management           |
|            Optimization              |
|                                      |
|   ----                                |
|  |    |                               |
|  |    |   ---                         |
|  |    |  /   \                        |
|  |    | |     |                       |
|  |____|  \___/                        |
|                                      |
|  Optimize Revenue Over 90 Days       |
|   - Adjust Prices and Quantities     |
|   - Consider Stock Constraints       |
|   - Manage Shipments                 |
|                                      |
\______________________________________/
```

## Problem Description

### Variables

- $Q_t$: Adjusted quantity forecast for day $t$

- $P_t$: Adjusted price for day $t$
- $Q_0$: Initial forecasted quantity sold for each day
- $P_0$: Initial price for each day
- $\alpha$: Correlation of price to quantity sold change
- $S_t$: Stock level on day $t$
- $S_{\min}$: Minimum stock required
- $T$: Number of days (90 in this case)
- $k$: Constant representing the allowed percentage change in price (e.g., 0.2 for 20%)
- $R_j$: Quantity of items arriving on day $j$ (shipment)
- $A_j$: Day $j$ when shipment arrives

## Equations and Constraints

### Objective Function

Maximize the sum of daily revenues over 90 days:

$$\text{Maximize} \quad \sum_{t=1}^{90} Q_t \times P_t$$

### Adjusted Quantity Forecast

The adjusted quantity forecast for each day $t$:

$$Q_t = Q_0 - \alpha(P_t - P_0)$$

### Stock Level

The stock level $S_t$ changes based on initial stock, shipments, and quantities sold:

$$S_t = S_{t-1} + \sum_{j=1}^{m} R_j \cdot \delta_{A_j,t} - Q_t$$

where $\delta_{A_j,t}$ is the Kronecker delta, which is 1 if $t = A_j$ and 0 otherwise.

### Stock Constraint

The stock level on any day $t$ should be greater than or equal to the minimum stock required:

$$S_t \geq S_{\min}$$

### Price Range Constraint

The adjusted price $P_t$ should lie within the range defined by $P_0$ and the constant $k$:

$$P_0(1-k) \leq P_t \leq P_0(1+k)$$

## Mathematical Formulation

Bringing it all together, we have the following formulation:

$$\text{Maximize} \quad \sum_{t=1}^{90} Q_t \times P_t$$

subject to:

$$Q_t = Q_0 - \alpha(P_t - P_0)$$

$$S_t = S_{t-1} + \sum_{j=1}^{m} R_j \cdot \delta_{A_j,t} - Q_t$$

$$S_t \geq S_{\min}$$

$$P_0(1-k) \leq P_t \leq P_0(1+k)$$

## Optimization Problem Nature

This optimization problem can be classified as a **quadratic programming problem**. The objective function involves a product of linear terms, leading to a quadratic term. Here is a more detailed explanation:

- The **objective function** $\sum_{t=1}^{90} Q_t \times P_t$ involves multiplying the adjusted quantity forecast by the adjusted price for each day. Given $Q_t = Q_0 - \alpha(P_t - P_0)$, substituting this into the objective function gives:

  $$\sum_{t=1}^{90} (Q_0 - \alpha(P_t - P_0)) \times P_t$$

  which simplifies to a quadratic form:

  $$\sum_{t=1}^{90} Q_0 P_t - \alpha P_t^2 + \alpha P_0 P_t$$

- The **stock constraint** $S_t = S_{t-1} + \sum_{j=1}^{m} R_j \cdot \delta_{A_j,t} - Q_t$ ensures that the stock level is maintained above a minimum threshold across the 90 days.

- The **price range constraint** $P_0(1-k) \leq P_t \leq P_0(1+k)$ ensures that the adjusted price remains within a reasonable range.

The combination of these elements makes the problem suitable for quadratic programming methods, though it can also be approached with general nonlinear optimization techniques due to the presence of the quadratic term in the objective function.

## Solution

To solve this problem, we use Python's optimization library `SciPy`. The approach involves defining the objective function, constraints, and using `scipy.optimize.minimize` to find the optimal prices.

### Python Solver

Here is the Python code to solve this modified optimization problem:

Listing 1: Python Code to Solve the Optimization Problem

```python
import numpy as np
from scipy.optimize import minimize

# Parameters
Q_0 = 100  # Initial forecasted quantity sold per day
P_0 = 10   # Initial price
alpha = 0.5  # Correlation of price to quantity sold change
S_0 = 1000  # Initial stock
S_min = 100  # Minimum stock required
T = 90  # Number of days
k = 0.2  # Allowed percentage change in price

# Shipments arriving on specific days
# Example: shipments = [(day, quantity), ...]
shipments = [(10, 200), (30, 300), (60, 150)]
shipment_days = [shipment[0] for shipment in shipments]
shipment_quantities = [shipment[1] for shipment in shipments]

# Bounds for P_t
P_min = P_0 * (1 - k)
P_max = P_0 * (1 + k)
bounds = [(P_min, P_max) for _ in range(T)]

# Objective function to maximize (negative of sum of revenues)
def objective(P):
    Q = Q_0 - alpha * (P - P_0)
    revenue = np.sum(Q * P)
    return -revenue  # Negative for minimization
```

```
# Constraint: stock level on each day should be >= minimum stock
def stock_constraint(P):
    S_t = S_0
    for t in range(T):
        Q_t = Q_0 - alpha * (P[t] - P_0)
        if t + 1 in shipment_days:
            S_t += shipment_quantities[shipment_days.index(t + 1)]
        S_t -= Q_t
        if S_t < S_min:
            return S_t - S_min
    return 0

# Constraints dictionary
constraints = {'type': 'ineq', 'fun': stock_constraint}

# Initial guess for P_t
P_initial = np.full(T, P_0)

# Solve the optimization problem
result = minimize(objective, P_initial, bounds=bounds, constraints=constraints)

# Check if the optimization was successful
if result.success:
    optimal_P = result.x
    optimal_Q = Q_0 - alpha * (optimal_P - P_0)
    total_revenue = np.sum(optimal_Q * optimal_P)
    print(f"Optimal Prices: {optimal_P}")
    print(f"Total Revenue: {total_revenue}")
else:
    print("Optimization failed:", result.message)
```

## Explanation

- **Objective Function**: The objective function calculates the negative of the sum of revenues over 90 days to convert the problem into a minimization problem (as required by `scipy.optimize.minimize`).
- **Adjusted Quantity Forecast**: The forecasted quantity for each day is adjusted based on the price for that day.
- **Stock Constraint**: Ensures that the sum of adjusted quantities over 90 days does not exceed the available stock minus the minimum required stock.
- **Price Range Constraint**: Ensures that the price for each day stays within the specified range.

- **Solver**: Uses `scipy.optimize.minimize` with bounds and constraints to find the optimal prices that maximize the total revenue.

This code provides a practical way to determine the optimal prices over a 90-day period while considering the constraints on stock and price range.