**Project No: 8**

**Project Title:**

**Prepared by: Rose Mary Jose**

**Date: 14-08-23**

Prepared for “Thoughtware Training Private Limited”, under Guidance of its CEO Mr. Pattabhi Raman. The project will be subject to further research, modification and exclusive use of “Thoughtware Training Private Limited”

**# Problem 1: Optimal Policy for Discrete Demand**

**Objective**

An optimal policy for discrete demand refers to the most effective strategy to manage inventory when the demand for items is countable and occurs in distinct units.

The objective of exploring the optimal policy for discrete demand in inventory management is to develop strategies that ensure efficient stock levels while minimizing costs and meeting customer demands. This has become necessary due to the inherent complexity of managing inventory in scenarios where demand occurs in distinct units. Traditional inventory management models designed for continuous demand may not be effective in such cases, leading to either excessive carrying costs or frequent stockouts.

An optimal policy for discrete demand offers numerous benefits for businesses. It helps businesses balance holding excess inventory with costs, reduces waste, and improves resource utilization. This approach aligns inventory levels with actual demand patterns, reducing waste and improving forecasting accuracy. Applications of this approach include retail, manufacturing, healthcare, and logistics, where retailers can optimize shelf replenishment strategies, manufacturers can avoid production halts due to parts shortages, and healthcare institutions can ensure critical medical supplies availability. Implementing the optimal policy for discrete demand leads to enhanced customer satisfaction, cost savings, and improved competitiveness and profitability.

**Problem statement**

A newsvendor purchases a number of copies of *"The Computer Journal”* (weekly). The observed demands during each of the last 52 weeks were:

[15,19,9,12,9,22,4,7,8,11,14,11,6,11,9,18,10,0,14,12,8,15,14,18,17,4,4,5,9,8,6,7,12,15,15,19,9,10,9,16,17,14,14,19,17,15,18,11,11,8,13,12]

Suppose each copy is purchased for 25 cents and sold for 75 cents, and his supplier pays him 10 cents for each unsold copy. calculate the optimal Quantity Q?

One common approach for addressing discrete demand is the "Newsvendor Model," which is a fundamental concept in inventory management. The Newsvendor Model considers the trade-off between ordering too much (resulting in excess inventory and associated costs) and ordering too little (leading to stockouts and potential revenue loss).

The procedure for finding the optimal solution to the newsboy problem when the demand is assumed to be discrete is a natural generalization of the continuous case. The optimal solution procedure is to locate the critical ratio between two values of F(Q) and choose the Q corresponding to the higher value. From the given data of 52 weeks, it is easy to calculate the mean and the standard deviation. From the given data the critical ratio can be calculated using the equation:

= F(Q\*)

where, = Cost per Unit of inventory remaining at the end of the period (known as the overage cost or cost of overstocking)

= Cost per Unit of unsatisfied demand (known as the underage cost or cost of understocking)

**Mathematical Formulation**

Suppose, the demand D is a continuous non negative random variable with the density function f(x) and the cumulative distribution function F(x). The decision variable Q is the number of units to be purchased at the beginning of the period.

The cost function G(Q) is given as,

G(Q) =

The optimal solution equation is

So, F(Q\*)

Where, = purchase cost – return cost

= sold cost – purchase cost

**Python Implementation**

Python code for finding the critical ratio is given below. Here a function calculate\_optimal\_stock\_level is defined using def function and the critical ratio is calculated using the formula for calculating F(Q\*). Later z-score is calculated using the ppf function from the stats library in python. Optimal stock level is calculated using the formula,

Q\* = mean + z \* standard deviation

def calculate\_optimal\_stock\_level(purchase\_price, selling\_price, refund, demand,understock\_cost, overstock\_cost, mean\_demand, std\_demand):  
 critical\_ratio = understock\_cost / (understock\_cost + overstock\_cost)  
  
 # Calculate the optimal stock level using the inverse CDF (percent point function) of the normal distribution  
 z = stats.norm.ppf(critical\_ratio)  
 optimal\_stock\_level = round(mean\_demand + z \* std\_demand)  
  
 return optimal\_stock\_level  
# Calculate the critical ratio  
purchase\_price = float(input('Enter the purchase price:'))  
selling\_price = float(input('Enter the selling price:'))  
refund = float(input('Enter the refund amount:'))  
demand = list(map(int, input("Enter demand values (separated by commas):").replace(' ', '').split(',')))  
  
understock\_cost = selling\_price - purchase\_price  
overstock\_cost = purchase\_price - refund  
mean\_demand = np.mean(demand)  
std\_demand = np.std(demand)  
  
optimal\_stock\_level = calculate\_optimal\_stock\_level(purchase\_price, selling\_price, refund, demand,  
 understock\_cost, overstock\_cost, mean\_demand, std\_demand)  
print("Optimal Stock Level:", optimal\_stock\_level)

The output obtained is:

Enter the purchase price:100  
Enter the selling price:150  
Enter the refund amount:20  
Enter demand values (separated by commas): 15,19,9,12,9,22,4,7,8,11,14,11,6,11,9,18,10,0,14,12,8,15,14,18,17,4,4,5,9,8,6,7,12,15,15,19,9,10,9,16,17,14,14,19,17,15,18,11,11,8,13,12  
Optimal Stock Level: 15

**Conclusion**

The Python code for calculating optimal stock levels for discrete demand provides valuable insights for inventory management in uncertain and discrete scenarios. It uses a critical ratio-based approach to determine the optimal stock level, combining costs of understocking and overstocking. The code uses statistical concepts like the standard normal distribution and inverse Cumulative Distribution Function to translate risk thresholds into actionable stock levels. The optimal stock level is suggested for decision-makers, enhancing cost-effectiveness and customer satisfaction by ensuring product availability. This is especially important in sectors where maintaining an optimal stock balance can significantly impact revenue and customer loyalty.

**Reference**

Supply Chain Analytics, T.A.S Vijayraghavan, Chapter 6- Inventory Concepts, Costs and Basic Models – 6.7 Optimal Stock Level in Newsvendor Models – optimal policy for discrete demand (example 6.5 Page no: 216)

**# Problem 2: EOQ in the All-Unit Discount Structure**

**Objective**

The objective of studying the Economic Order Quantity (EOQ) model within an all-unit discount structure is to explore the optimization of inventory management in scenarios where suppliers offer price breaks for different order quantities. This has become necessary due to the prevalence of quantity discounts in supply chains, influencing businesses to make informed decisions about order quantities.

By understanding EOQ in the context of all-unit discount structures, businesses can determine the most cost-effective order quantities that consider both ordering and carrying costs as well as changing unit prices. This knowledge empowers organizations to strike a balance between minimizing inventory costs and maximizing operational efficiency.

Benefits of delving into this topic include enhanced cost savings through optimal order quantity selection, improved inventory turnover, reduced carrying costs, and better supplier negotiations. By integrating EOQ with all-unit discount considerations, businesses can make well-informed decisions that lead to improved supply chain management, streamlined operations, and competitive advantage in a dynamic market environment.

**Problem Statement**

Suppose, a supplier offers the slabs of quantities with the respective prices for those slabs stated below:

|  |  |
| --- | --- |
| Quantity Slabs | Price per unit |
| Up to 500 | $100 |
| between 500 and 1000 | $95 |
| 1000 and more | $90 |

Assume the supplier offers the All-Unit Discount structure to a buyer. Assume annual Demand is 1000 units, ordering cost S is $100, and h is 20% (0.20). Determine the appropriate quantity to exploit the discount.

**Mathematical Formulation**

The model developed by Ford Harris for finding the optimum order quantity has become known as the basic EOQ model and it serves as the basis for many of the pull inventory policies currently used in practice. The basic EOQ formula is developed from a total cost equation involving the trade-off between ordering or procurement cost and inventory-carrying cost as it assumes a constant price for the units produced, which keeps the material cost constant irrespective of the quantity ordered. It is expressed as

Total Cost (TC) = Ordering Cost (OC) +Inventory-carrying Cost (ICC) +Material Cost (MC)

Where, *TC* is the total annual relevant inventory cost(dollars), *Q* is the size of each order to replenish inventory (units), *D* is the annual demand for the item in inventory(units), *S* is ordering cost, *C* is the value of the item carried in inventory, and *h* is the carrying cost as a percent of the item value.

The term *D/Q* represents the number of times orders are placed to its supply source. The term *Q/2* is the average amount of inventory on hand referred to as cycle stock.

As *Q* varies, the ordering cost decreases and the inventory- carrying cost increases. It can be shown mathematically that an optimal order quantity (*Q\*)* exists where the two costs are in balance.

To obtain the optimal *Q\**, differentiate the formula of total cost with respect to Q and equates to zero:

EOQ,

**Python Implementation**

Step 1: Evaluate the order quantity for each price , 0i

def calculate\_eoq(D, S, h, quantity\_ranges, price\_ranges):  
 eoq\_values = []

for q\_range, price in zip(quantity\_ranges, price\_ranges):  
 q1, q2 = q\_range  
 ci = price  
  
 if q2 == float('inf'):  
 q2 = D # Set q2 to the annual demand if it's infinity  
 eoq = math.sqrt((2 \* D \* S) / (h \* ci))  
 eoq\_values.append((q1, q2, eoq))  
  
 return eoq\_values

Step 2: optimal order quantity for each price is selected, which follows three scenarios:

1. , then set
2. , then set
3. , this scenario can be ignored as it is considered for

Step 3: for each *i*, calculate the total cost of ordering units using the equation, *TC*

# Calculate total cost for each case using the formula TC = (D/optimalEOQ)\*S + (optimalEOQ/2)\*C\*h + D\*Ci  
total\_costs = []  
  
for q\_range, optimal\_eoq, price in zip(quantity\_ranges, optimal\_eoq\_values, price\_ranges):  
 q1, q2 = q\_range  
 ci = price  
  
 total\_cost = (D / optimal\_eoq) \* S + (optimal\_eoq / 2) \* ci \* h + D \* ci  
 total\_costs.append(total\_cost)

Step 4: for overall *i*, select order quantity with the lowest cost .

QuantityRange Price Q\*(EOQ based on criteria) Optimal EOQ Total Cost  
0 (0, 500.0) 100 100.0 100.0000 102000.0

1 (501, 1000.0) 95 500.0 102.597783 99950.0

2 (1001, inf) 90 1000.0 105.409255 99100.0

Hence, placing Q = 1000 will be optimal to exploit the discount and also to minimize the total cost.

**Conclusion**

The Economic Order Quantity (EOQ) model, combined with Python programming, provides a valuable framework for optimizing inventory management decisions. By calculating the optimal order quantity (Q) considering various price tiers, businesses can minimize Total Cost (TC) and efficiently navigate quantity discounts. This approach facilitates cost savings and streamlines operations by avoiding overstocking and understocking issues. By integrating EOQ and all-unit discount considerations, businesses can make informed choices that balance ordering costs, carrying costs, and fluctuating unit prices, enhancing supply chain efficiency, financial performance, and supplier negotiations. As the business landscape evolves, the EOQ model remains a powerful tool for inventory management and operational success.

**Reference**

Supply Chain Analytics, T.A.S Vijayraghavan, Chapter 6- Inventory Concepts, Costs and Basic Models – 6.10 Quantity Discount EOQ Models -EOQ in the All-Unit Discount Structure (subtopic: 6.10.1 Page no: 225)

**# Problem 3: EOQ in Marginal-Unit Discount Structure**

**Objective**

The Objective is to decide the optimal quantity that will minimize the total cost(annual) consisting of ordering, inventory-carrying, and material costs. As for All-Unit Discount model here too, the procedure is to evaluate the optimal lot size for each marginal price .

The study examines the application of EOQ principles in marginal unit discount frameworks to optimize order quantities and minimize total costs. By strategically selecting order quantities aligned with changing price tiers, businesses can achieve substantial cost savings. This knowledge leads to efficient inventory turnover, reduced carrying costs, and improved supplier negotiation outcomes. This knowledge equips businesses with a competitive edge, enabling informed decisions to optimize supply chain efficiency and operational performance.

**Problem Statement**

Consider the table given below.

|  |  |
| --- | --- |
| Quantity Slabs | Marginal- Unit Price |
| First 100 | $500.5 |
| Next 100 | $490.5 |
| 200 and above | $475.5 |

Given, *D* =1200, and *S* = 25 with *h* = 0.20. Determine the optimal quantity that will minimize the annual total cost consisting of ordering, inventory-carrying and material costs.

**Mathematical Formulation**

These models are also referred to as Incremental-Unit Discount structure models. Here, the pricing schedule contain specified break points , where = 0 and the prices are denoted as , for 0i. In these models, the marginal cost of a unit decreases with the break points as compared to the average cost in the case of the All-Unit Discount structure models.

Let be the material cost of placing an order of units for each value of *i*, 0i. Assume that = 0. Then,

Let us assume that an order quantity of *Q* in the range of and units are placed. That is, . The material cost of this order quantity *Q* can be calculated as

. The total annual cost components are derived as follows

Ordering Cost =

Inventory-Carrying Cost =

Material Cost =

So, the total annual cost will be,

The optimal lot size for price is obtained by minimizing the total cost with respect to *Q* and equating to zero. The calculus gives the optimal lot size for the price .

for the price =

This formula is similar to the EOQ formula, but the fixed cost of ordering has increased from *S* to , that is an additional cost of due to quantity discounts.

Determining the optimal lot size is quite similar to the steps that are followed in that case of All-Unit Discount structure and is outlined in the python implementation.

**Python Implementation**

Step 1: calculating the values.

def calculate\_Vi(q\_values, ci\_values):  
 Vi\_values = [0] # Initialize with V0 = 0  
 for i in range(1, len(q\_values)):  
 delta\_q = q\_values[i] - q\_values[i - 1]  
 Vi = delta\_q \* ci\_values[i - 1] + Vi\_values[i - 1]  
 Vi\_values.append(Vi)  
 return Vi\_values

Step 2: lot size for price is obtained.

def calculate\_EOQ(D, S, Vi, qi, Ci, h):  
 return math.sqrt((2 \* D \* (S + Vi - qi \* Ci)) / (h \* Ci))

Step 3: optimal lot size is determined

1. , then set
2. , then set
3. , then set

Step 4: For each *i*, total cost of ordering is determined

# Calculate costs and add them to the data dictionary  
ordering\_costs = [(D / EOQ) \* S for EOQ in optimal\_EOQ\_values]  
inventory\_carrying\_costs = [(Vi + (EOQ - qi) \* Ci) \* (h / 2) for EOQ, qi, Ci, Vi in zip(optimal\_EOQ\_values, q\_values, ci\_values, Vi\_values)]  
material\_costs = [(D \* (Vi + (EOQ - qi) \* Ci)) / EOQ for EOQ, qi, Ci, Vi in zip(optimal\_EOQ\_values, q\_values, ci\_values, Vi\_values)]  
total\_costs = [ordering + carrying + material for ordering, carrying, material in zip(ordering\_costs, inventory\_carrying\_costs, material\_costs)]

The output is:

D S Ci Vi Qi Optimal Qi Ordering Cost \  
0 1200.0 25.0 500.5 0.0 0 24 1250.000000   
1 1200.0 25.0 490.5 50050.0 100 158 189.873418   
2 1200.0 25.0 475.5 99100.0 200 319 94.043887   
 Inventory Carrying Cost Material Cost Total Cost   
0 1201.20 600600.000000 603051.200000   
1 7849.90 597050.704225 604234.710127   
2 15568.45 587442.105263 601309.515831

From the output it is clear that the optimal quantity will be to place an order for 319 where the total cost is minimum.

**Conclusion**

The Economic Order Quantity (EOQ) model, incorporating marginal unit discounts, offers valuable insights into optimizing inventory management practices. By considering marginal unit discounts in EOQ calculations, businesses can make more accurate and cost-efficient order quantity determinations. This approach allows businesses to strategically align order quantities with changing price tiers, resulting in cost savings through reduced carrying costs and optimal stock levels. Integrating the EOQ model with marginal unit discount considerations enhances supply chain efficiency, financial performance, and supplier relationships, providing organizations with a competitive edge for informed decisions, maximizing profitability in a dynamic market environment.

**Reference**

Supply Chain Analytics, T.A.S Vijayraghavan, Chapter 6- Inventory Concepts, Costs and Basic Models – 6.10 Quantity Discount EOQ Models -EOQ in Marginal-Unit Discount Structure (6.10.2) (example 6.8 Page no: 228)

**# Problem 4: EOQ models with Backorders (Planned shortages)**

**Objective**

The objective of this study is to explore the Economic Order Quantity (EOQ) models within the context of incorporating backorders. This exploration is vital due to the evolving dynamics of supply chains, where stockouts can lead to missed opportunities and dissatisfied customers.

The study explores EOQ principles for accommodating backorders, minimizing costs and avoiding stockouts. By analysing EOQ models with backorders, businesses can determine optimal order quantities, ensuring steady supply, enhanced inventory turnover, reduced holding costs, and better resource utilization. This approach leads to more efficient operations, increased profitability, and a competitive advantage in today's market landscape.

**Problem Statement**

Suppose that the annual demand *D* = 10,000 units, *S* = $500, *H* = $10, = $40. Determine and compare the Total Cost when a backorder situation exist and when such a situation doesn’t exist.

**Mathematical Formulation**

Let *Q* be the lot size to be ordered, *B* be the shortages or backorder quantity, be the shortage or backorder cost per unit of time short, *H* be the inventory-carrying cost, and maximum inventory, *M = Q-B*. The total cost calculation should now include the backorder cost, ordering cost, and inventory-carrying cost.

Total Cost = Annual Ordering cost + Inventory-Carrying Cost of Average Inventory of *M* + Inventory-Holding Cost of Backorder *B*.

Here, refers to the proportion of the cycle time where positive inventory *M = Q-B* exists and refers to the time when the negative inventory (backorders) exists.

Differentiating with respect to *B* and *Q*, respectively and equating them to zero, we get the expressions for B and Q as follows:

**Python Implementation**

Step1: *Q* and *B* values are obtained using their respective equations

def calculate\_QB(D, S, H, back\_ordercost):  
 Q = round(math.sqrt((2 \* D \* S) / H) \* math.sqrt((H + back\_ordercost) / back\_ordercost))  
 B = round(Q \* (H / (H + back\_ordercost)))  
 return Q, B

Step 2: is calculated after substituting the values of *B* and *M* values.

def calculate\_M\_TC(Q, B, H):  
 M = Q - B  
 TC = (D / Q) \* S + (M / 2) \* H \* (M / Q) + (B / 2) \* back\_ordercost \* (B / Q)  
 return M, TC

Step 3: to compare the total cost if such a backorder situation doesn’t exist and no backorder were planned. In this case, the normal EOQ *Q*\* is obtained and total annual cost is calculated using the formula.

def calculate\_EOQ(D, S, H):  
 Q\_eoq = round(math.sqrt((2 \* D \* S) / H))  
 return Q\_eoq

# Calculate EOQ and corresponding total cost  
Q\_eoq = calculate\_EOQ(D, S, H)  
TC\_eoq = (D / Q\_eoq) \* S + (Q\_eoq / 2) \* H

The output obtained is:

Backorder scenario:  
Q value (Q): 1118  
Backorder quantity (B): 224  
Total cost (TC): 8944.28  
  
EOQ scenario:  
EOQ value (Q): 1000  
Total cost (TC): 10000.0  
Difference in total costs (Original - Backorder): -1055.7245080500888

We see that the planned shortage in this case has saved $10,000 – 8944.27 = $ 1055.73 even assuming a slightly higher backorder cost of $40 as against the holding cost of $10.

**Conclusion**

The Economic Order Quantity (EOQ) model and Python programming provide a comprehensive approach to inventory management decisions. By determining Total Cost (TC) for scenarios with and without backorders, the analysis reveals the importance of considering backorders in inventory planning. Implementing planned shortages can yield substantial cost savings, with a calculated difference of $1055.73 between the two scenarios. This quantitative basis for decision-making enhances financial performance and ensures smoother operations, ultimately improving overall supply chain efficiency.

**Reference**

Supply Chain Analytics, T.A.S Vijayraghavan, Chapter 6- Inventory Concepts, Costs and Basic Models – 6.10 Quantity Discount EOQ Models -EOQ Models with Backorders (6.10.3) (example 6.9 Page no: 230)

**# Problem 5: Conditions of Uncertainty: Discrete Distributions**

**Objective**

The objective of this topic is to find the optimal quantity of order and the total cost of that order if the demand is uncertain. Demand fluctuations are very difficult to understand, and sophisticated statistical tools are required to systematically study demand patterns and measure uncertainty. From statistical theories we know demand can be represented as a random variable, which may be discrete or a continuous random variable.

This topic is crucial for businesses to manage uncertain variables like demand, customer preferences, and market trends. Exploring discrete distributions helps quantify and manage these uncertainties, improving risk assessment and resource allocation. This knowledge enhances strategic planning, forecasting, budgeting, inventory management, and ultimately leads to informed decisions, risk mitigation, and opportunities seizing, ultimately contributing to improved competitiveness and overall success.

**Problem Statement**

We will assume the following situation of certainty initially and analyse the situation when it is uncertain.

Suppose the following information is given:

Annual Demand, D = 3600 units, C = $100, h = 25% (0.25). H = $25, S = $200

Assume the probability distribution of demand during lead time and the possible units of inventory short or excess during the lead time with various reorder point is given as a table.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Annual demand | probability | 100 | 110 | 120 | 130 | 140 | 150 | 160 |
| 100 | 0.01 | 0 | 10 | 20 | 30 | 40 | 50 | 60 |
| 110 | 0.06 | -10 | 0 | 10 | 20 | 30 | 40 | 50 |
| 120 | 0.24 | -20 | -10 | 0 | 10 | 20 | 30 | 40 |
| 130 | 0.38 | -30 | -20 | -10 | 0 | 10 | 20 | 30 |
| 140 | 0.24 | -40 | -30 | -20 | -10 | 0 | 10 | 20 |
| 150 | 0.06 | -50 | -40 | -30 | -20 | -10 | 0 | 10 |
| 160 | 0.01 | -60 | -50 | -40 | -30 | -20 | -10 | 0 |

**Mathematical Formulation**

The demand and the probability distribution along with the possible units of inventory during the lead time with various reorder points are given. From the table it is observable that the numbers above the diagonal elements are excess (excess inventory) and below the diagonal are shortages (opportunity losses). By applying the same concept of decision making under uncertainty in statistics with the respective demand probabilities a new table is obtained. From that table we summarize the total excess and shortages and compute the respective costs associated with the excesses and shortages. Excesses refers to carrying excess inventory, and we use the given inventory carrying cost for calculating the cost. The stock-out cost is used for shortage cost. Total annual shortage cost and total annual cost is calculated. The reorder level which has the minimum total cost is found and we can model that situation to determine the optimal using the total cost expression:

Total Cost = Ordering Cost + Inventory Carrying Cost (Cycle Stock) + Inventory Carrying Cost (Excess) + Shortage (Stock-out Cost)

Where D, annual demand, S is Ordering Cost, H is carrying Cost, e is excess inventory, g is the shortage, and is the shortage cost.

Solving this by differentiating w.r.t to Q and equating to zero, we get

**Python Implementation**

Enter annual demand (D): 3600  
Enter setup cost (S): 200  
Enter carrying cost (h): 0.25  
Enter unit price (C): 100  
Q: 240.0  
Number of cycles: 15.0  
Total cost: 6000.0

From the given information the EOQ and the total cost is calculated for the situation where demand is certain.

Unit of Excess and Shortages Table:  
 Reorder Points | 100 110 120 130 140 150 160   
----------------|----------------------------  
 100 | 0.00 0.10 0.20 0.30 0.40 0.50 0.60   
 110 | -0.60 0.00 0.60 1.20 1.80 2.40 3.00   
 120 | -4.80 -2.40 0.00 2.40 4.80 7.20 9.60   
 130 | -11.40 -7.60 -3.80 0.00 3.80 7.60 11.40   
 140 | -9.60 -7.20 -4.80 -2.40 0.00 2.40 4.80   
 150 | -3.00 -2.40 -1.80 -1.20 -0.60 0.00 0.60   
 160 | -0.60 -0.50 -0.40 -0.30 -0.20 -0.10 0.00

Units of excess and shortages calculated by multiplying the numbers with their respective probabilities.

# Calculate total excess (e) and total shortage (g) values  
total\_shortage\_values = [abs(sum(unit\_excess\_and\_shortage[j][i] for j in range(i + 1, len(reorder\_points)))) for i in range(len(demand\_values))]  
total\_excess\_values = [abs(sum(unit\_excess\_and\_shortage[j][i] for j in range(i)) + unit\_excess\_and\_shortage[i][i]) for i in range(len(demand\_values))]

For each demand their respective total excess and shortages are calculated by adding up the excess of each demand and shortages of each demand separately. In the data matrix the off-diagonal entries below are shortages and those above are excesses. Then the excess cost and shortage cost are calculated for each demand.

# Calculate excess cost (E) and shortage cost per cycle (G)  
H = h\*C # Expected carrying cost  
back\_ordercost = float(input("Enter shortage cost per unit of time short (pi): "))  
excess\_cost\_values = [e \* H for e in total\_excess\_values]  
shortage\_cost\_values = [g \* back\_ordercost for g in total\_shortage\_values]  
total\_annual\_shortage\_costs = [shortage\_cost \* cycles for shortage\_cost in shortage\_cost\_values]

# Calculate total annual cost for each demand  
total\_annual\_costs = [shortage\_cost + excess\_cost for shortage\_cost, excess\_cost in zip(total\_annual\_shortage\_costs, excess\_cost\_values)]

Excess cost is calculated as, inventory carrying cost \* excess (e) and the shortage cost per cycle (G) is calculated as, stock-out cost \* shortage (g). Then the total annual shortage cost G(D/Q) is calculated and then the total annual cost is calculated which is the summation of total annual shortage cost and excess cost. Here the annual demand is considered for the calculation of annual shortage cost.

Demand Total Excess (e) Excess Cost (E) Total Shortage (g) \  
0 100 0.0 0.0 30.0   
1 110 0.1 2.5 20.1   
2 120 0.8 20.0 10.8   
3 130 3.9 97.5 3.9   
4 140 10.8 270.0 0.8   
5 150 20.1 502.5 0.1   
6 160 30.0 750.0 0.0

Shortage Cost per Cycle Total Annual Shortage Cost Total Annual Cost   
0 300.0 4500.0 4500.0   
1 201.0 3015.0 3017.5   
2 108.0 1620.0 1640.0   
3 39.0 585.0 682.5  
4 8.0 120.0 390.0   
5 1.0 15.0 517.5   
6 0.0 0.0 750.00

From the above data frame, the minimum total cost is found and the reorder point, excess cost, and total shortage of that total cost is used for determining the optimal and the total cost is calculated.

Lowest Total Annual Cost Row:  
Demand 140.0  
Total Excess (e) 10.8  
Excess Cost (E) 270.0  
Total Shortage (g) 0.8  
Shortage Cost per Cycle (G) 8.0  
Total Annual Shortage Cost 120.0  
Total Annual Cost 390.0  
Name: 4, dtype: float64  
Q\*: 245  
Total Annual Cost (Lowest): 6389

It is noticeable that the total has increased from $6000 to $6389, an increase of $6389, which is about 6.5% more. Thus, we can conclude the following:

The following costs will rise to cover the uncertainty: 1) stock-out costs 2) inventory carrying costs of safety stock

The result may or may not be significant

**Conclusion**

The study of discrete probability distributions offers valuable insights into modelling and managing uncertainties in business contexts. It uncovers effective tools for analysing and quantifying uncertain variables, enabling informed decision-making in complex and dynamic environments. This topic is relevant to real-world scenarios like demand forecasting, financial planning, and risk assessment. By understanding discrete distributions, businesses can make more accurate predictions, optimize resource allocation, and formulate robust strategies. This leads to improved risk management, enhanced operational efficiency, and opportunities capitalized amidst uncertainty. By navigating uncertain terrain confidently, organizations can achieve better outcomes and a competitive edge in today's evolving business landscape.

**Reference**

Supply Chain Analytics, T.A.S Vijayraghavan, Chapter 7- Inventory under Uncertainty and Service Levels – 7.3 Understanding Demand Uncertainty -Conditions of Uncertainty: discrete distributions (7.3.1) (example 7.1 Page no: 240)

**# Problem 6: Service Levels and Product Availability Measures**

**Objective**

The objective of the problem is to determine the effectiveness of the Order Up To policy in managing inventory levels and meeting customer demand while minimizing lost sales and stock-out situations.

This objective is crucial to strike a balance between inventory costs and customer service levels. By analysing the performance of the Order Up To policy, the company can ensure product availability for customers while minimizing excess inventory costs and stockouts. This policy enables a dynamic order quantity that adapts to demand fluctuations, enhancing efficiency. In the supply chain context, the Order Up To policy optimizes inventory management by addressing variable demand patterns. Unlike fixed order quantity models like Economic Order Quantity (EOQ), this policy adapts orders based on leftover inventory at the end of each cycle. It prevents frequent stockouts, improves customer satisfaction, and helps avoid excess inventory costs. This approach is particularly valuable in industries with uncertain demand, allowing effective resource allocation and enhanced supply chain performance.

**Problem Statement**

For the sample data given we will look at an order up to policy of keeping 45 units at the beginning of every order cycle. Here each period, that is, cycle is assumed to be 2 weeks. The vendor starts with an inventory of 45 units and meets the demand that comes to him every period. If at the end of the cycle, 5 units are left over, then he orders another 40 units so that he starts the next order cycle with the 45 units again. This is often referred to as Order Up To policy where the order quantity may change depending on the items left over at the end of every cycle unlike fixed order quantity models like EOQ. Table is created with columns to reflect the lost sales and stock-out situations if the demand is maintained and the Order Up To policy is followed.

**Mathematical Formulation**

Service level represents the probability of meeting customer demand without stockouts. It quantifies the reliability of an inventory system's ability to satisfy customer orders promptly, often expressed as a percentage. Higher service levels indicate greater assurance of product availability but may require higher inventory investment. In inventory policies, service levels can be classified into two types: Unit Service Level (USL) and Order Service Level (OSL).

Order Service Level (OSL): it can be interpreted as the proportion of cycles where customer demand was satisfied. The OSL represents the probability of not having a stock-outs during the placement of an order. If there are many order cycles or orders that may exist during a year, the stock-out probability times the number of cycles provides the probability of not having stock-outs during the year. So, if there are cycles left without the customer demand be satisfied in such cycles a stock-out can be expected and thus there comes the term OSOR as order stock-out risk which is 1-OSL.

Unit Service Level (USL): it indicates the percentage of units of demand filled during any period, whereas the unit stock-out risk (USOR) specifies the quantities of units unfilled or short during that period. USL also known as product availability is defined as,

Product Availability (USL) = 1 -

USOR = 1 – USL

**Python Implementation**

period demand quantity\_sold quantity\_left lost\_sales \  
0 1 37.0 37.0 8.0 0.0   
1 2 37.0 37.0 8.0 0.0   
2 3 75.0 45.0 0.0 30.0   
3 4 73.0 45.0 0.0 28.0   
4 5 53.0 45.0 0.0 8.0   
  
 stock\_out\_during\_period   
0 0   
1 0   
2 1   
3 1   
4 1

We take user inputs for the desired range of periods and a constant keeping unit value, which represents the inventory level to be maintained at the beginning of each order cycle. Then a data frame with columns representing each period's demand, quantity sold, quantity left, lost sales, and stock-out occurrences is created. Through a loop, we inputted demand values for each period. It calculates various metrics such as the quantity sold (limited by the keeping unit), quantity left (inventory remaining after sales), lost sales (difference between demand and quantity sold), and whether a stock-out occurred during each period. The Data Frame is updated with these calculated values.

The Order Up To policy is modelled by dynamically adjusting order quantities based on inventory levels. It helps analyse inventory performance, sales, and stock-outs over the specified period range.

def calculate\_metrics(df):  
 sum\_demand = df['demand'].sum()  
 sum\_lost\_sales = df['lost\_sales'].sum()  
 sum\_stock\_out = df['stock\_out\_during\_period'].sum()

print(f"Sum of Demand: {sum\_demand}")  
 print(f"Sum of Lost Sales: {sum\_lost\_sales}")  
 print(f"Sum of Stock Out During Period: {sum\_stock\_out}")

total\_stock\_out\_situvation = total\_period - sum\_stock\_out  
 order\_service\_level = total\_stock\_out\_situvation / total\_period

OSOR = 1 - order\_service\_level  
 USL = 1 - (sum\_lost\_sales / sum\_demand)

return sum\_demand, sum\_lost\_sales, sum\_stock\_out, total\_stock\_out\_situvation, order\_service\_level, OSOR, USL

Then calculates the various performance metrics based on the provided data frame. It computes the sum of demand, lost sales, and stock-out occurrences. Then, it derives the total instances of periods without stock-outs and calculates the order service level and its complementary measure (OSOR). Then computes the Unit Service Level (USL) as the ratio of fulfilled demand to total demand.

total\_stock\_out\_situvation: 30  
order\_service\_level: 0.6  
OSOR: 0.4  
USL: 0.8553677932405567

**Conclusion**

In conclusion, this provides valuable insights into the efficiency and performance of an inventory management strategy utilizing the Order Up To policy. By calculating metrics such as total demand, lost sales, stock-out occurrences, order service level, OSOR, and Unit Service Level (USL), the code provides a comprehensive evaluation of the inventory system's ability to meet customer demand and mitigate stock-outs. This analysis aids in making informed decisions to optimize inventory levels, strike a balance between customer satisfaction and operational costs, and enhance overall supply chain performance. Ultimately, the code facilitates effective inventory management and contributes to informed decision-making within the realm of supply chain operations.

**Reference**

Supply Chain Analytics, T.A.S Vijayraghavan, Chapter 7- Inventory under Uncertainty and Service Levels – 7.4 Service Levels and Product Availability- 7.4.1 and 7.4.2 (example 7.2 Page no: 247)

**# Problem 7: Determining Reorder Point when Demand is Uncertain**

**Objective**

The objective of this problem is to calculate the Economic Order Quantity (EOQ), safety stock, and reorder point for an item in order to optimize inventory management. By determining these key parameters, the goal is to strike a balance between minimizing costs related to inventory holding and replenishment while ensuring sufficient stock availability to meet customer demand during lead times.

Safety stock and reorder points are crucial in inventory management, ensuring organizations maintain adequate inventory to meet customer demand and handle supply chain uncertainties. These strategies minimize stockout risks, enhance customer satisfaction, and improve supply chain performance by ensuring smooth operations and consistent customer service.

**Problem Statement**

Consider the following details of an item: Monthly demand, d = 11000 units, standard deviation of demand, = 3500 units, Unit cost of the item C = $0.15, Lead time, LT = 1.5 months, acquisition cost for processing an order, S = $20, In-stock probability, P = 0.75, assuming a normal distribution for the demand during the lead time. Inventory carrying cost percentage, h = 0.25. Determine the EOQ, safety stock, and the reorder point for this item.

**Mathematical Formulation**

From lead time demand and variability when lead time are multiple periods, we derive the expression for the reorder level when it is assumed that the demand during the lead time follows a normal distribution. We assume demand during any period follows a normal distribution with parameters (*d,* ), that is mean and standard deviation respectively, and the lead time is denoted as LT periods or weeks.

Then it follows that the demand during the lead time LT also follows a normal distribution with parameters and

Where, and with a given OSL.

ROP for the given OSL is

This implies that

ROP =

Here the additional component added to account for demand variability during the lead time. Hence it is evident that this is the extra inventory kept in order to handle demand fluctuations in addition to the average stock during the lead time. It can be therefore interpreted that this expression refers to safety stock we have to maintain to address demand fluctuations. Hence, we have

That simplifies ROP as,

SS

**Python Implementation**

The Python code calculates inventory management metrics like Economic Order Quantity (EOQ), safety stock, and reorder point based on input parameters. It uses normal distribution and formulas to determine critical values and helps users balance inventory costs, lead time, and service levels in supply chain operations.

def metrics(d,Sd,C,LT,S,P,h):  
 EOQ = round(np.sqrt((2 \* d \* S) / (h / 12 \* C)))  
 z = round(norm.ppf(P),2)  
 safety\_stock = round(z \* Sd \* np.sqrt(LT))  
 d\_prime = round(d \* LT)  
 sd\_prime = round(Sd \* np.sqrt(LT))  
 ROP =round(d\_prime + safety\_stock)  
 return EOQ, z, safety\_stock, d\_prime, sd\_prime, ROP

The output obtained is:

EOQ: 11866  
z: 0.67  
Safety Stock: 2872  
Reorder Point (ROP): 19372

**Conclusion**

The problem at hand involves determining the Economic Order Quantity (EOQ), safety stock, and reorder point for an item within inventory management. EOQ helps strike a balance between holding and replenishment costs. Safety stock acts as a cushion against demand and lead time uncertainties. Reorder point indicates when to reorder to avoid stockouts. These metrics are vital to optimize inventory levels and enhance supply chain performance. By calculating these values, businesses can avoid excess inventory costs while ensuring product availability to meet customer demand. This approach is instrumental in achieving efficient inventory management and maintaining customer satisfaction within the supply chain network.

**Reference**

Supply Chain Analytics, T.A.S Vijayraghavan, Chapter 7- Inventory under Uncertainty and Service Levels – 7.4 - Service Levels and Product Availability – Determination of Reorder Point when Demand is Uncertain (7.4.3) (example 7.3 Page no: 250)

**# Problem 8: What-if-Scenarios in Inventory Decisions**

**Objective**

The objective of this problem is to analyse the inventory management situation to determine the safety stock, order service level (OSL), and Unit Service Level (USL). These metrics help assess the adequacy of inventory to meet customer demand and understand the performance of the inventory system.

This problem addresses inventory management critical aspects affecting supply chain efficiency. Identifying safety stock, order service level, and Unit Service Level helps businesses meet customer demand while minimizing stockouts and costs. Quantifying risk and performance align inventory levels with demand variability, enhancing operational resilience and balancing cost-efficiency and service quality.

**Problem Statement**

Suppose for an item, the weekly demand is d = 3000 units and the standard deviation of the weekly demand is = 600 units. Assume the lead time Lt is given as 2 weeks. If an order of 9000 units is placed when there are 7000 units in the inventory, what will be the safety stock for this item and how much OSL is maintained for this situation? What will be the USL be for this item?

**Mathematical Formulation**

For demand less than the reorder point, the stock-out quantity is zero. However, for demand greater than the reorder point, the stock-out quantity is the demand minus the reorder point. This is often referred to as the unit normal loss (UNL) function where f () is a normal probability distribution function and F () is the cumulative distribution function for standard normal distribution.

Unit Normal Loss (UNL) =

The expected number of units out of stock annually = E(z)

From this USL can be calculated as,

USL = 1 -

= 1 -

**Python Implementation**

To solve this problem, first, calculate the safety stock by using the standard formula involving the standard deviation of weekly demand and lead time. Determine the z-score corresponding to the desired service level using the standard normal distribution table. Then, calculate the order service level (OSL) using the cumulative distribution function of the standard normal distribution. Next, compute the Unit S-ervice Level (USL) by dividing the safety stock by the product of the weekly demand and lead time. These calculations help quantify the buffer needed to handle demand and lead time uncertainties, assess the probability of stockouts during lead time, and evaluate how effectively demand is met. This approach aids in informed inventory management decisions, ensuring product availability and enhancing supply chain performance.

def calculate(d,Sd,LT,ROP,order\_quantity):  
 # Calculate d' and sd'  
 d\_prime = d \* LT  
 sd\_prime = round(Sd \* np.sqrt(LT))  
  
 # Calculate safety stock from ROP equation  
 safety\_stock = ROP - d\_prime  
  
 # Calculate z from safety stock and sd'  
 z = safety\_stock / sd\_prime  
  
 # Calculate E(z)  
 pdf = norm.pdf(z)  
 cdf = norm.cdf(z)  
 unl = pdf - z \* (1 - cdf)  
  
 # Calculate USL  
 usl = 1 - (unl \* sd\_prime / order\_quantity)  
 return d\_prime, sd\_prime, safety\_stock, z, unl, usl

Safety Stock: 1000.00 units  
z: 1.18  
Unit Normal Loss (UNL): 0.0587  
USL: 0.9945

**Conclusion**

In conclusion, this problem underscores the pivotal role of safety stock, OSL, and USL in inventory management for robust supply chain operations. By determining these metrics, organizations can mitigate uncertainties in demand and lead time, ensuring optimal inventory levels that meet customer expectations. This approach fosters efficient resource allocation, reduces stockouts, and enhances customer satisfaction. In the broader supply chain context, these metrics aid in risk management, seamless operations, and strategic decision-making. Embracing these principles empowers businesses to strike a harmonious balance between inventory costs and service levels, fostering resilience and excellence in the supply chain ecosystem.

**Reference**

Supply Chain Analytics, T.A.S Vijayraghavan, Chapter 7- Inventory under Uncertainty and Service Levels – 7.6 -Estimation of USL or Fill rate – Estimation of Number of Units out of stock or shortage in an order cycle (7.6.1) (example 7.4 Page no: 253)

**# Problem 9: Back Order Case**

**Objective**

The objective is to find the optimal inventory policy (order quantity and reorder point) that balances the costs of holding excess inventory and potential stockouts, aiming to minimize total inventory costs. The EOQ model and related calculations help in finding this optimal balance between holding excess inventory and incurring stockout costs.

The calculations consider factors such as demand variability, desired service levels, and holding costs. Achieving this optimal balance enhances operational efficiency by minimizing inventory-related expenses while ensuring adequate stock availability to meet customer demand. This problem is necessary for businesses to manage their inventory efficiently, reduce holding costs, and avoid stockouts, ultimately improving customer satisfaction and overall financial performance.

**Problem Statement**

Assume that *D* = 15000, *S* =1000, *H* = 30, = 300, = 50 and *OSL* = 0.70. Then determine the order quantity and reorder point. Also, determine the imputed cost .

**Mathematical Formulation**

In the backorder case, the cost of increasing ROP by one more unit will be *H(Q/D)*. However, if we do not add an additional unit, the backorder penalty cost of is charged with the stock-out probability in a cycle, which is OSOR.

Therefore, the marginal cost of adding a unit in ROP = the marginal cost of not adding a unit to ROP. So,

H(Q/D) = OSOR

this gives,

OSOR = HQ/ D

or = HQ/OSORD

where, the EOQ and ROP calculated as,

**Python Implementation**

Step 1: calculating the EOQ.

# calculate the quantity  
 Q = math.sqrt((2 \* D \* S) / H)

Step 2: Calculating ROP and SS. For calculating the z value is calculated from the OSL value given. With that z value unit normal loss E(z) is also calculated.

# Calculate ROP and SS  
 z = round(norm.ppf(OSL), 2)  
 ROP = d\_prime + (z \* Sd\_prime)  
 SS = z \* Sd\_prime  
  
 ## calculating E\_z from Z value  
 cdf = norm.cdf(z)  
 pdf = norm.pdf(z)  
 E\_z = round(pdf - z \* (1 - cdf), 2)

Step 3: calculating the order stock out risk from OSL value given

USL = round(1 - ((E\_z \* Sd\_prime) / S), 2)  
USOR = round(1 - USL, 2)

## calclate order stock out risk (OSOR)  
OSOR = round(1 - OSL, 3)

Step 4: the value of backorder penalty is calculated as follows:

## calcualte pi value  
 pi = (H \* Q) / (OSOR \* D)

The output obtained is:

Quantity (Q) : 1000.0  
Z-score : 0.52  
Reorder Point (ROP) : 326.0  
Safety Stock (SS) : 26.0  
Unit Normal Loss Value (E\_z) : 0.19  
Unit Service Level (USL) : 0.99  
Unit Stock Out Risk (USOR) : 0.01  
Order Stock Out Risk (OSOR) : 0.3  
Backorder Penalty Cost (pi) : 6.666666666666667

**Conclusion**

The inventory management problem emphasizes the importance of achieving an optimal balance between holding excess inventory and mitigating stockout risk. By calculating critical inventory metrics like Economic Order Quantity (EOQ), reorder point (ROP), and service and risk levels, businesses can effectively manage their inventory systems and financial resources. The goal is to streamline operations by minimizing cumulative costs associated with carrying inventory and stockouts, boosting cost-efficiency, customer satisfaction, and overall organizational performance.

The problem provides an actionable framework for decision-makers to adopt proactive inventory strategies, offering an empirically informed foundation for ensuring optimal stock levels and reducing unnecessary holding costs. This aligns with supply chain management principles, promoting prudent resource allocation and responsiveness to dynamic market demands. Aligning inventory practices with this objective strengthens organizations' competitive edge and achieves sustainable growth in a volatile business landscape.

**Reference**

Supply Chain Analytics, T.A.S Vijayraghavan, Chapter 7- Inventory under Uncertainty and Service Levels – 7.8 – Backorder Case (Page no: 257)

**# Problem 10: Lost Sales Case**

**Objective**

The objective of this problem is to determine the optimal order quantity (Q) and associated parameters to minimize the total costs associated with inventory management while meeting certain service level requirements. Through this project the optimal order quantity (Q\_star) and related parameters that minimize the total costs associated with inventory management, including carrying costs, setup costs, and backorder costs are determined, while meeting a specified service level requirement (USL). This is crucial for businesses to efficiently manage their inventory, reduce carrying costs, and meet customer demand consistently. By considering factors like demand variability and service level targets, the solution helps businesses strike a balance between stock availability and cost efficiency. This optimization holds significant benefits, including improved cash flow, reduced storage expenses, and enhanced customer satisfaction through timely order fulfilment. It finds practical application in industries like retail, manufacturing, and distribution, optimizing operations and maximizing profitability.

**Problem Statement**

Assume that *D* = 15000, *S* =1000, *H* = 30, = 300, = 50 and *OSL* = 0.70. Then determine the optimal order quantity when stock-out cost is not explicitly given.

**Mathematical Formulation**

In the case of lost sales, the stock out cost will include the opportunity loss of revenue besides customer dissatisfaction. the marginal analysis also yields slightly different results because of the impact of lost sales on inventory levels. Here, adding one unit to ROP will cost H(Q/D) minus OSOR H(Q/D), which is the carrying cost of the unused inventory when one unit of safety stock is being carried. and if we do not add the unit, the penalty in the event of stock-out includes .

suppose D = 50, Q = 25 if a stock-out occurs in the first cycle, it means that 26 units have been demanded while only 25 units have been supplied. hence when we start the next cycle, we will have the full Q= 25 on hand. with D = 50, only 24 units are to be satisfied in the second cycle. this would make us carry one more unit additionally in the second cycle.

The marginal cost of adding one additional unit to the ROP = marginal cost of not adding.

From that we can calculate the OSOR with value given.

OSOR =

From that OSL, Safety stock and USL are calculated. While the USL is approximately the same the safety stock in the case of lost sales is more than the backorder case. From the preceding discussion it is clear that we will be able to determine various unknowns of which a few are known with respect to service levels and stock-out costs. Here the Q value is calculated using the formula,

Python Implementation

The optimal order quantity (Q\_star) and associated metrics in a dynamic inventory management are calculated. Input necessary parameters like demand, carrying cost, setup cost, d\_prime, sd\_prime, and the desired Unit Service Level (USL) optimal quantity is calculated. The iteration process follows a convergence-based approach, calculating the initial Q value and the Unit Service Level (UNL). The main loop iterates to refine the optimal order quantity by solving for the z-score, which is obtained from the UNL value. OSL and Order Service Out of Rate (OSOR) is calculated using the z-value. The backorder cost is calculated using the refined order quantity, and the formula is adjusted until the change in order quantity falls below a convergence threshold or the maximum iteration limit is reached. The code outputs the final optimal order quantity, excess probability, and backorder cost.

Q = math.sqrt((2 \* D \* S) / H)  
  
# Calculate USOR and E\_z  
USOR = 1 - USL  
E\_z = (Q \* USOR) / sd\_prime  
iteration = 0  
  
# Convergence criterion  
convergence\_threshold = 0.001  
  
while True:  
 # Step 3: Solve for z  
 equation\_to\_solve = lambda z: stats.norm.pdf(z) - z \* (1 - stats.norm.cdf(z)) - E\_z  
 result = optimize.root(equation\_to\_solve, 0.0)  
 z\_value = result.x[0]  
  
 # Step 4: Calculate new Q and back order cost  
 OSL = stats.norm.cdf(z\_value)  
 OSOR = 1 - OSL  
 back\_ordercost = (H \* Q) / (OSOR \* D)  
 Q\_star = math.sqrt((2 \* D \* (S + back\_ordercost \* E\_z \* sd\_prime)) / H)  
  
 # Check for convergence  
 if abs(Q - Q\_star) < convergence\_threshold or iteration >= 100:  
 break  
  
 Q = Q\_star  
 E\_z = (Q \* USOR) / sd\_prime  
 iteration += 1

From the output it is clear that for a USL of 0.95 the optimal quantity will be 1065 units. Similarly, the optimal quantity for any USL value can be calculated using the code used here.

Enter the demand (D): 15000  
Enter the carrying cost (H-whole number): 30  
Enter the setup cost (S-whole number): 1000  
Enter d\_prime (whole number): 300  
Enter sd\_prime(whole number): 50  
Enter the USL(in decimal): 0.70  
Final Q\_star: 1581.137481516645  
Final E\_z: 9.486819494825857  
Final Back Order Cost: 3.162273164941952

**Conclusion**

In this dynamic inventory management scenario, the optimization process yielded a final optimal order quantity (Q\_star) of approximately 1066 units. The associated excess probability (E\_z) was determined to be around 1.066, reflecting a refined estimate based on the desired Unit Service Level (USL) of 0.95. Additionally, the calculated backorder cost was found to be approximately 2.55. This analysis underscores the dynamic nature of inventory optimization, where iterative adjustments to order quantities are made to align with desired service levels and minimize costs. The refined Q\_star and associated metrics provide insights into achieving a balance between inventory management and operational efficiency while meeting customer service expectations.

**Reference**

Supply Chain Analytics, T.A.S Vijayraghavan, Chapter 7- Inventory under Uncertainty and Service Levels – 7.9 – Lost Sales Case (Page no: 250)