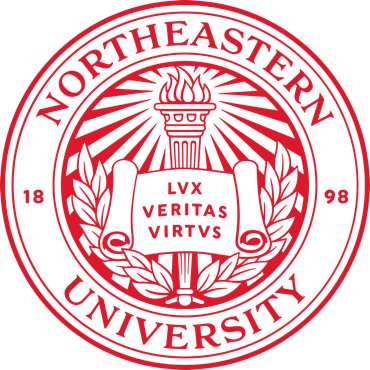


Module 3

A prescriptive model for strategic decision-making, An inventory Management Decision Model



###### **ALY6050 21597 - Introduction to Enterprise Analytics**

###### **Instructor: Prof. Roy Wada**

###### **Date: May 7, 2022**

###### **By: Sakshi Mehta**

**Abstract**

It is very important for inventories to be managed well as they have to represent the investment strategy for all the organizations. Excess of it or not getting enough of it can be problematic for organization as it will lead to bad management of financial and operations.

However, to figure this problem out managers have to make decision as to when the additional inventories should be produced and when the order should be placed so as to minimize the total inventory cost which includes other variable and fixed cost like holding cost, ordering cost etc.

This report shows how these fundamental inventory decisions can be made and optimized using the data models in Excel and R programming language.

**Introduction**

We will make the inventory model by first making the data model and adding and defining all the fixed and variable costs. Given we have the annual demand of 15000 units and the cost of each unit is 80$, holding cost accounts up to 18% of the unit value and a total of $220 for each order that is placed with the supplier.

**Analysis**

**PART 1**

(i) Starting with defining all the parameters that will influence the total inventory cost. The given information is first stated.

|  |  |  |
| --- | --- | --- |
| Annual Order | 15000 |  |
| cost per unit | 80 |  |
| opportunity/holding cost | 18% of cost per unit | 14.4 |
| cost per order | 220 |  |

(ii) Performing mathematical operations to produce the annual ordering and annual holding cost and EOQ (Economic order quantity) by using the mathematical formula as follows :

**EOQ = SQRT((2 \* ORDERING COST \* DEMAND)/ HOLDING COST)**

|  |  |
| --- | --- |
| Economic Order Quantity (EOQ) | 677.0032004 |

**Annual ordering cost = (Annual demand \* Ordering cost) / EOQ**

**Annual Holding cost = (holding cost \* EOQ) / 2**

|  |  |
| --- | --- |
| Annual Ordering | 4874 |
| Annual Holding | 48774 |

(iii) Creating the data model. This data model consists of order quantities and holding cost, ordering cost and total cost with respect to the EOQ. Total cost is equal to Holding cost plus Ordering cost.

|  |  |  |  |
| --- | --- | --- | --- |
| DATA TABLE |  |  |  |
| EOQ | Holding cost | Ordering Cost | Total Cost |
| 500 | 3600 | 6600 | 10200 |
| 550 | 3960 | 6000 | 9960 |
| 600 | 4320 | 5500 | 9820 |
| 650 | 4680 | 5076.923077 | 9756.923077 |
| 670 | 4824 | 4925.373134 | 9749.373134 |
| 700 | 5040 | 4714.285714 | 9754.285714 |
| 750 | 5400 | 4400 | 9800 |
| 800 | 5760 | 4125 | 9885 |

(iv) Now, using the model to find the quantity that results in the smallest total cost

Table

Description automatically generated

We, see that out of all the EOQ values the one with 670 shows the minimum total cost which is equal to 9749.

(v) plotting of total cost VS EOQ

The graph depicts the same result, we see that total cost is minimum at around 650.

(vi) Now, using excel solver to see the test results that we got above and verify it.

A picture containing timeline

Description automatically generated

Here, we see that excel solver gives us the value as 677. This is the exact EOQ where total cost in going to be minimum.

(vi) Conducting what-if analysis in Excel to see the difference in total cost by changing model parameters by using the two-way tables.

Table

Description automatically generated

The first vertical column shows the various Economic Order Quantity and the first row shows the different cost per unit. Therefore, by changing these 2 parameters the table shows the change in total cost.

# Installing the required libraries.

library("SCperf")  
library("triangle")  
library("Rmisc")

## Loading required package: lattice

## Loading required package: plyr

# —— PART 1——-

# Defining model parameters and the given information.

Annual demand (D)

annual\_demand = 15000

Unit cost (C)

each\_unit\_cost = 80

Holding cost(H)

holding\_cost = 0.18 \* each\_unit\_cost

Ordering cost(S)

order\_cost = 220

# Calculating EOQ given the total annual demand, ordering cost and holding cost.

EconomicOrderQuantity <- EOQ(annual\_demand,order\_cost,holding\_cost)  
EconomicOrderQuantity

## Q T TVC   
## 677.003 0.045 9748.846

We see the value of EOQ as 677.

Calculating annual ordering cost

annual\_ordering\_cost <- annual\_demand/EconomicOrderQuantity[1]\*order\_cost  
annual\_ordering\_cost

## Q   
## 4874

Calculating annual holding cost

annual\_holding\_cost <- EconomicOrderQuantity[1]/2\*holding\_cost  
annual\_holding\_cost

## Q   
## 4874

# Calculating total cost (annual ordering cost + annual holding cost)

total\_inventory\_cost <- (annual\_demand/EconomicOrderQuantity[1]\*order\_cost)+(EconomicOrderQuantity[1]/2\*holding\_cost)  
total\_inventory\_cost

## Q   
## 9749

# optimizing EOQ value

intervalEOQ <- c(500,750) #setting the interval

Defining the function

fEOQ <- function(EconomicOrderQuantity) {  
 (EconomicOrderQuantity[1]/2\*holding\_cost)+(annual\_demand/EconomicOrderQuantity[1]\*order\_cost)  
}

# Using the optimise function to get the optimal EOQ value.

optimize\_EOQ <- optimise(f=fEOQ , interval= intervalEOQ, lower=min(intervalEOQ), upper = max(intervalEOQ), maximum = FALSE, tol=.Machine$double.eps^0.5)  
optimize\_EOQ

## $minimum  
## [1] 677  
##   
## $objective  
## [1] 9749

We found out that for the cost to minimum the optimal EOQ value should be 677 and the cost for that will be 9749.

# Printing the required EOQ and Min Total Cost

print(paste("Optimized EOQ is:", round(optimize\_EOQ[[1]], 0)))

## [1] "Optimized EOQ is: 677"

print(paste("Minimum Total Cost is:", round(optimize\_EOQ[[2]], 0)))

## [1] "Minimum Total Cost is: 9749"

# —– Part 2 ——

# Storing the given minimum, maximum and mode of annual demand.

min\_annual\_demand = 13000  
max\_annual\_demand = 17000  
mode\_annual\_demand = 15000

# Calculating EOQ for each 1000 demand values.

demand\_range = round(rtriangle(n=1000, min\_annual\_demand, max\_annual\_demand, mode\_annual\_demand))  
Reorder\_append\_values <- c()  
for (val in demand\_range)  
{  
 Reorder\_level = round(sqrt((2\*val\*order\_cost)/holding\_cost))  
 Reorder\_append\_values <- c(Reorder\_append\_values, Reorder\_level)  
}  
print(Reorder\_append\_values)

## [1] 668 690 667 671 688 676 673 679 682 648 666 648 666 674 689 651 683 703  
## [19] 650 678 647 715 695 682 678 655 650 720 680 692 659 660 670 662 666 667  
## [37] 679 680 694 712 691 702 678 658 667 657 671 672 669 677 653 676 661 656  
## [55] 703 684 683 646 713 689 667 685 664 686 694 659 708 662 658 679 686 678  
## [73] 672 660 657 675 664 678 657 676 671 653 683 679 673 636 681 659 664 669  
## [91] 658 671 672 679 703 718 685 700 653 679 703 678 681 665 690 707 703 672  
## [109] 664 667 691 706 705 674 667 667 660 675 658 690 658 703 706 667 700 680  
## [127] 681 703 720 655 684 668 688 710 662 677 679 714 701 677 697 717 693 678  
## [145] 665 716 685 695 689 651 695 644 666 669 670 706 658 672 694 681 705 674  
## [163] 700 657 663 662 705 717 661 683 672 700 704 703 657 694 678 699 678 689  
## [181] 670 679 673 653 659 710 679 687 688 686 715 688 648 661 691 647 693 649  
## [199] 681 710 687 681 668 680 659 679 681 664 665 696 708 681 714 655 676 655  
## [217] 713 691 673 652 695 657 715 651 666 677 663 672 703 681 700 650 683 691  
## [235] 639 709 673 681 689 641 691 677 703 674 704 684 677 672 638 705 677 675  
## [253] 654 697 684 678 655 674 671 673 714 679 704 664 687 700 661 642 676 640  
## [271] 688 679 673 671 670 695 673 643 678 688 679 665 702 693 689 680 642 670  
## [289] 690 675 664 679 667 662 649 701 649 660 692 692 699 673 699 695 685 684  
## [307] 672 679 684 679 694 709 674 671 706 681 671 669 648 659 676 698 674 680  
## [325] 686 668 686 695 661 667 689 689 646 682 680 691 671 642 649 662 664 642  
## [343] 691 714 648 702 698 690 673 700 670 674 676 697 673 662 643 673 673 680  
## [361] 646 706 686 699 698 655 685 661 687 657 694 671 642 664 663 704 651 673  
## [379] 704 671 703 641 679 705 684 664 655 648 679 674 676 704 667 695 691 690  
## [397] 671 670 672 715 682 679 657 677 651 685 674 665 684 659 691 666 681 670  
## [415] 638 670 695 678 697 675 708 658 675 666 650 666 684 672 656 643 710 675  
## [433] 677 658 648 679 671 680 673 667 670 663 670 673 669 663 659 675 673 681  
## [451] 695 690 687 700 687 704 686 684 641 647 661 637 645 698 711 702 718 684  
## [469] 673 669 684 673 665 645 635 653 674 692 661 711 647 679 678 675 674 669  
## [487] 698 665 684 661 669 703 691 708 688 666 680 683 673 706 633 647 686 673  
## [505] 642 652 680 667 708 688 684 657 667 692 660 665 666 673 681 637 700 673  
## [523] 680 701 673 701 692 654 707 702 692 683 663 641 665 685 664 645 668 702  
## [541] 676 670 677 671 672 699 692 675 681 678 678 660 671 696 689 697 688 680  
## [559] 652 668 678 668 702 660 673 669 686 668 712 675 669 672 659 655 699 700  
## [577] 719 696 686 686 695 670 698 697 662 670 692 651 662 683 680 654 652 696  
## [595] 642 667 670 656 639 692 691 680 683 689 681 705 675 697 690 651 697 698  
## [613] 689 696 669 676 681 676 665 688 676 703 646 658 695 670 665 667 648 670  
## [631] 677 682 660 682 646 643 702 656 660 665 662 636 660 683 654 675 703

# Calculating holding cost for each EOQ.

each\_holding\_cost\_append <- c()  
for (i in Reorder\_append\_values)  
{  
 each\_holding\_cost = i\*holding\_cost/2  
 each\_holding\_cost\_append <- c(each\_holding\_cost\_append, each\_holding\_cost)  
}

each\_holding\_cost\_append

## [1] 4810 4968 4802 4831 4954 4867 4846 4889 4910 4666 4795 4666 4795 4853  
## [15] 4961 4687 4918 5062 4680 4882 4658 5148 5004 4910 4882 4716 4680 5184  
## [29] 4896 4982 4745 4752 4824 4766 4795 4802 4889 4896 4997 5126 4975 5054  
## [43] 4882 4738 4802 4730 4831 4838 4817 4874 4702 4867 4759 4723 5062 4925  
## [57] 4918 4651 5134 4961 4802 4932 4781 4939 4997 4745 5098 4766 4738 4889  
## [71] 4939 4882 4838 4752 4730 4860 4781 4882 4730 4867 4831 4702 4918 4889  
## [85] 4846 4579 4903 4745 4781 4817 4738 4831 4838 4889 5062 5170 4932 5040  
## [99] 4702 4889 5062 4882 4903 4788 4968 5090 5062 4838 4781 4802 4975 5083  
## [113] 5076 4853 4802 4802 4752 4860 4738 4968 4738 5062 5083 4802 5040 4896  
## [127] 4903 5062 5184 4716 4925 4810 4954 5112 4766 4874 4889 5141 5047 4874  
## [141] 5018 5162 4990 4882 4788 5155 4932 5004 4961 4687 5004 4637 4795 4817  
## [155] 4824 5083 4738 4838 4997 4903 5076 4853 5040 4730 4774 4766 5076 5162  
## [169] 4759 4918 4838 5040 5069 5062 4730 4997 4882 5033 4882 4961 4824 4889  
## [183] 4846 4702 4745 5112 4889 4946 4954 4939 5148 4954 4666 4759 4975 4658  
## [197] 4990 4673 4903 5112 4946 4903 4810 4896 4745 4889 4903 4781 4788 5011  
## [211] 5098 4903 5141 4716 4867 4716 5134 4975 4846 4694 5004 4730 5148 4687  
## [225] 4795 4874 4774 4838 5062 4903 5040 4680 4918 4975 4601 5105 4846 4903  
## [239] 4961 4615 4975 4874 5062 4853 5069 4925 4874 4838 4594 5076 4874 4860  
## [253] 4709 5018 4925 4882 4716 4853 4831 4846 5141 4889 5069 4781 4946 5040  
## [267] 4759 4622 4867 4608 4954 4889 4846 4831 4824 5004 4846 4630 4882 4954  
## [281] 4889 4788 5054 4990 4961 4896 4622 4824 4968 4860 4781 4889 4802 4766  
## [295] 4673 5047 4673 4752 4982 4982 5033 4846 5033 5004 4932 4925 4838 4889  
## [309] 4925 4889 4997 5105 4853 4831 5083 4903 4831 4817 4666 4745 4867 5026  
## [323] 4853 4896 4939 4810 4939 5004 4759 4802 4961 4961 4651 4910 4896 4975  
## [337] 4831 4622 4673 4766 4781 4622 4975 5141 4666 5054 5026 4968 4846 5040

# Calculating annual total cost

annual\_total\_cost <- annual\_ordering\_cost\_each + each\_holding\_cost\_append

# Calculating total number of orders annually

total\_no\_orders <- demand\_range / Reorder\_append\_values

# Creating data frame

inventory\_data <- data.frame(demand\_range, Reorder\_append\_values, each\_holding\_cost\_append, annual\_ordering\_cost\_each, annual\_total\_cost, total\_no\_orders)  
inventory\_data

## demand\_range Reorder\_append\_values each\_holding\_cost\_append  
## 1 14606 668 4810  
## 2 15600 690 4968  
## 3 14558 667 4802  
## 4 14720 671 4831  
## 5 15478 688 4954  
## 6 14934 676 4867  
## 7 14805 673 4846  
## 8 15103 679 4889  
## 9 15202 682 4910  
## 10 13732 648 4666  
## 11 14526 666 4795  
## 12 13728 648 4666  
## 13 14496 666 4795  
## 14 14882 674 4853  
## 15 15543 689 4961  
## 16 13855 651 4687  
## 17 15265 683 4918  
## 18 16161 703 5062  
## 19 13818 650 4680  
## 20 15059 678 4882  
## 21 13680 647 4658  
## 22 16710 715 5148  
## 23 15819 695 5004  
## 24 15240 682 4910  
## 25 15038 678 4882  
## 26 14056 655 4716  
## 27 13823 650 4680  
## 28 16968 720 5184  
## 29 15134 680 4896  
## 30 15685 692 4982  
## 31 14213 659 4745  
## 32 14256 660 4752  
## 33 14680 670 4824  
## 34 14352 662 4766

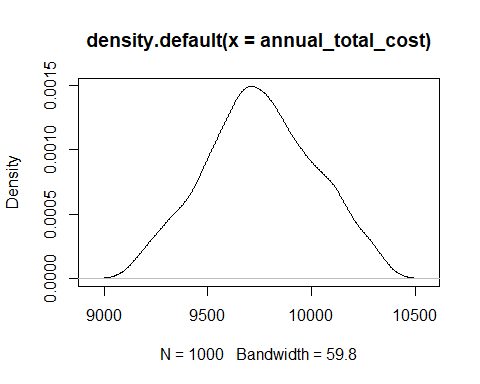
# Calculating confidence intervals and the type of probability distribution.

for total annual cost, we found out that the confidence interval is 9736 to 9769.

CI(annual\_total\_cost, ci = 0.95)

## upper mean lower   
## 9769 9753 9736

p1 <- density(annual\_total\_cost)  
plot(p1)



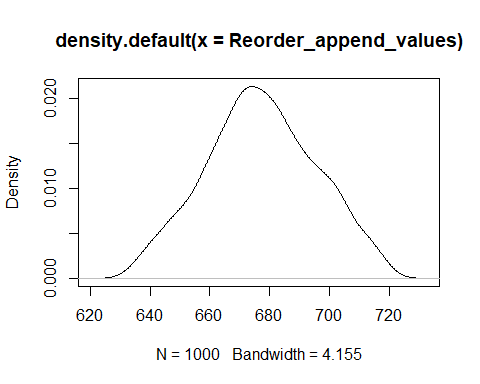
The probability distribution is normal.

#for EOQ, the confidence interval is 676 to 678

CI(Reorder\_append\_values, ci=0.95)

## upper mean lower   
## 678 677 676

p2 <- density(Reorder\_append\_values)  
plot(p2)



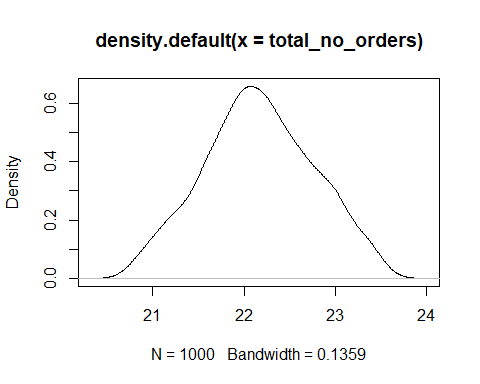
EOQ plot follows a normal probability distribution.

# for number of orders annually the confidence interval is 22 to 22.

CI(total\_no\_orders, ci= 0.95)

## upper mean lower   
## 22 22 22

p3 <- density(total\_no\_orders)   
plot(p3)



The curve follows a normal probability distribution.

**Conclusion**

Optimization of EOQ is required and a must calculation for organization to work properly in terms of finance. It is possible to get the best value and best count to order so that the total cost in minimum.

**Reference**

<https://www.wallstreetmojo.com/eoq-formula/>

https://rpubs.com/kimbrown345/286189