A I I E Transactions
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/uiie19

A Dynamic Lot Sizing Algorithm with Capacity Constraints

P. S. Eisenhut [a]
[a] IBM System Products Division, East Fishkill Facility , Hopewell Junction, New York, 12533

Published online: 09 Jul 2007.

To cite this article: P. S. Eisenhut (1975) A Dynamic Lot Sizing Algorithm with Capacity Constraints, A I I E Transactions, 7:2, 170-176, DOI: 10.1080/05695557508974999

To link to this article: http://dx.doi.org/10.1080/05695557508974999

A Dynamic Lot Sizing Algorithm with Capacity Constraints

P. S. EISENHUT
SENIOR MEMBER, AIIE
IBM System Products Division
East Fishkill Facility
Hopewell Junction, New York 12533

Abstract: A new algorithm has been developed to calculate economic lot sizes for manufactured products. The algorithm allows for fluctuating and uncertain demand patterns which limit the effectiveness of the classical Wilson EOQ formula. In an environment of uncertain future demand, its performance is on a par with the Wagner-Whitin technique. Yet, the computations required are considerably less. For a single product the algorithm would produce results identical to the Part/Period formula of IBM. However, a new derivation is provided which parallels the derivation of the Silver and Meal formula. A new dimension is added when the algorithm is extended to the situation where there are limits on the combined production rate of all products. The algorithm prevents production from exceeding the limit. It therefore allows lot sizing to be implemented gradually, and it continues to protect the production line against sudden increases in demand after implementation. Finally, the dynamic lot sizing algorithm with capacity constraints is compatible with techniques which may be required to provide additional production smoothing.

The Problem

Lot sizing is the process of determining how much of each product to make and when to make it. Classical lot sizing algorithms treat each product independently and assume that demands and costs are static. In fact, total production for all products may depend on limited capacity. In addition, demands and costs are always changing. A dynamic lot sizing algorithm is presented here which considers the production of all products simultaneously, and which responds to changing demands.

The objective of lot sizing is to combine requirements into orders in an economical fashion, without exceeding a limit on total capacity in any one time period. Matrix "A" in Table 1 is a "requirements matrix." It represents the quantity of each product that must be made in that time period, or in earlier time periods, if all commitments are to be met. The development of the requirements matrix presumes the existence of systems to keep track of inventory by product and to forecast demand by product by time period. Matrix "B" in Table 1 is an "order matrix." It contains what is actually planned to be made for each

Table 1: 3 Product example of the problem.

		PLANNING HORIZON →												UNCERTAINTY →
		TIME PERIOD												
		1	2	3	4	5	6	7	8	9	10	11	12	
	1	5	3	7	30	10	12	10	15	6	9	3	2	
PRODUCT	2	0	5	2	8	30	13	2	17	4	12	6	10	
	3	0	23	34	18	0	16	29	18	18	12	9	15	

(A) STARTS REQUIREMENTS MATRIX

	1	55	0	0	0	0	57	0	0	0	0	0	0
PRODUCT	2	0	15	0	0	45	0	0	39	0	0	0	10+
	3	0	23	34	34	0	0	47	0	39	0	0	15+
TOTAL		55	38	34	34	45	57	47	39	39	0	0	25+
CAPACITY		60	60	60	60	60	60	60	50	40	30	30	30

(B) PLANNED ORDER SIZE MATRIX

The problem is to economically convert a starts requirement matrix into a solutioned order size matrix without exceeding capacity in any one period.

Received November 1974; revised February 1975.

product in each time period. The lot sizing process grouped requirements from multiple time periods into common orders.

A lot sizing procedure to convert the requirements matrix into the order matrix should minimize total expected cost whenever possible. Total cost is the sum of the fixed costs per order, including setup, and the inventory cost of production not immediately required. Ordering each product in every time period would result in no inventory cost but excessive ordering cost. On the other hand, ordering each product only once during the planning horizon would result in excessive inventory cost. In addition, the procedure should not order more in any one time period than the production capacity. Each product has different requirements and different costs. These requirements fluctuate from period to period. It may not be possible, therefore, to order an "economical lot size" for every product in a given period without exceeding capacity. The lot sizing procedure must determine how much of each product to make in one period and how much to make in future periods. The procedure must not only take into consideration the relative costs of ordering versus costs of inventory for each product, but also the relative costs and requirements between products.

Evaluation of Costs

To compare alternate solutions, we will illustrate the method used here to compute cost. Refer to product #1 in Table 1 and compute the costs associated with the planned order size matrix for 12 periods. A setup (order) is made in period 1 and another in period 6. Suppose each order costs $100. Then the total setup, or ordering cost, is $200 for 12 periods.

The inventory cost of product #1 will be the inventory cost associated with the first order, plus the inventory cost associated with the second order. When computing inventory cost we will assume that requirements in the same period as they are ordered will not incur inventory cost. Thus in the first order there is no inventory cost for the 5 units required in period 1. Likewise, in the second order there is no charge for the 12 units required, and also ordered, in period 6. The inventory cost associated with the first order will be the cost of holding 3 units for 1 period, 7 units for 2 periods, 30 units for 3 periods, and 10 units for 4 periods. Likewise, in the second order there will be 10 units held for 1 period, 15 units held for 2 periods, 6 units held for 3 periods, 9 units held for 4 periods, 3 units held for 5 periods, and 2 units held for 6 periods. Mathematically the inventory cost for an order cycle, starting in period 1, is expressed as follows:

$$I(T) = h \cdot \sum^{T} (t-1)F(t). \qquad (1)$$

"I (T)" is the inventory cost associated with an order cycle of length T. "F (t)" is the requirements in period t. "h" is the cost of holding 1 unit for 1 period (see Appendix A). If we assume h = $1 then the inventory cost for product #1 in this example will be:

1st Order $I\,(T) = 3 + 2\,(7) + 3\,(30) + 4\,(10)$

$I\,(T) = \$147$

2nd Order $I\,(T) = 10 + 2\,(15) + 3\,(6) + 4\,(9) + 5\,(3) + 6\,(2)$

$I\,(T) = \$121$

Thus the total cost for product #1 for 12 periods, given the solution shown in Table 1, is $200 ordering cost, plus $268 inventory cost, or $468.

Two Classical Approaches

Two classical lot sizing procedures are the Wilson square root EOQ formula [1] and the Wagner-Whitin dynamic programming algorithm [2]. The EOQ formula is shown in Appendix B. The technique is very simple to use, but, unless the average requirements per period are fairly constant, it is not very accurate. To visualize the problem consider a requirements rate of 10 per period during the first 6 periods, and a requirements rate of 1000 during the next 6 periods of the planning horizon. The average rate of 505 will be used in the Wilson formula. The calculated order size will result in too much inventory cost during the first 6 periods and too much ordering cost during the second 6 periods of the plan.

The Wagner-Whitin algorithm will give an optimal solution when requirements are forecasted far into the future with certainty. The algorithm therefore provides a valuable benchmark against which to measure the performance of other techniques. However, in reality, forecasts often cover only short planning horizons. The Wagner-Whitin algorithm implicitly assumes that beyond the planning horizon, requirements are zero. The procedure finds a solution which optimizes cost over the planning horizon only. If at the time when the second order is to be placed a new requirements forecast is made, it will differ from the original. Requirements beyond the original planning horizon will no longer be zero. Table 2 illustrates the effect on total cost of assuming zero requirements after 6 periods each time an order size is determined.

Table 2: Comparison Wagner-Whitin with 6-period horizon.

	TIME PERIOD (T)									
	1	2	3	4	5	6	7	8	9	COST
REQUIREMENT F (T)	38	79	67	10	33	53	17	26	125	
OPTIMAL	194	0	0	0	129	0	0	0	125	$1158
WAGNER-WHITIN	117	0	110	0	0	96	0	0	125	$1224

Shows that Wagner–Whitin is not optimal when future requirements are uncertain (h = $1, S = $250).

Usually the accuracy of a forecast decreases progressively with distance into the future. The second forecast will thus be better than the first forecast even for the overlapping span of time. With the Wagner-Whitin technique the cost performance of the first order is often sacrificed for the sake of overall optimality over a longer horizon. However, if future requirements turn out to be different than anticipated, the overall optimality is lost. Future orders do not provide the cost performance originally anticipated when the first order size was determined. As the ordering process continues–forecast to forecast–it results in a long term solution consisting of a string of "first orders," each having higher than optimal costs. For this reason the Wagner-Whitin solution is not necessarily optimal and is not necessarily better than other approaches.

The reason that the Wagner-Whitin technique is not widely used is that it requires a great deal of computation time. It is necessary to compute the cost for many alternate solutions over the planning horizon. The number of computations increases proportionally to the number of discrete time periods and the number of products. In a system having a 6-month planning horizon divided into weeks, and a couple of hundred products, the cost of computer time could be prohibitive.

The approach described in this paper is intended for the following situation: where the forecasted requirements fluctuate from period to period, and are increasingly uncertain in the future; and where there is a need to gain significant economic benefit, without the computations themselves becoming too costly. Finally, the two classical procedures do not attempt to deal with more than one product at a time. The approach described here is for the situation where the combined production of several products is limited.

Dynamic Lot Sizing
Single Product

For a single product without capacity constraint, Silver and Meal suggest a strategy of minimizing the cost per period of each successive order cycle [3]. Cost per period for the first order is given by the equation:

$$C(T) = [S + I(T)]/T. \tag{2}$$

The first term, S, is the fixed cost per order, and the second term, $I(T)$, is the inventory cost. T is the time until the second order. The average rate of change for this per period cost is of interest.

$$C'(T) = [I'(T)/T] - [I(T)/T^2] - (S/T^2). \tag{3}$$

Equation (3) is the derivative of (2) with respect to T. For small values of T, $C'(T)$ is large and negative, indicating a large decrease in per period cost for each increment in T. As T increases, however, the per period cost decreases at a lesser and lesser rate and finally begins to increase. The point at which the per period cost stops decreasing is the point at which $C'(T) = 0$. An approach then is to substitute $C'(T) = 0$ into Eq. (3) and solve for $T = T^*$, the number of periods of requirements to be pulled into the order. Rather than do this directly, however, Silver and Meal first make an approximation to Eq. (3) "to allow ease of solution." A similar approximation is made here, but for a different reason:

$$\begin{aligned} I'(T) &= h(T-1)F(T) = 2h \sum^{T} (t-1)F(T)/T \\ &\approx 2h \sum^{T} (t-1)F(t)/T = 2I(T)/T. \end{aligned} \tag{4}$$

As before $F(T)$ is the requirements at period T. However, in this case the terms are stated in reverse order, and as discrete summations rather than integrals. In Eq. (3), rather than replace $I(T)$, as did Silver and Meal, we will replace $I'(T)$. The resulting equation is

$$C'(T) \approx I(T)/T^2 - S/T^2. \tag{5}$$

If we now let $C'(T) \to 0$ and solve for $T = T^*$ we get a very simple solution:

$$I(T^*) \to S. \tag{6}$$

If T is discrete, T^* is such that $I(T^*) \leqslant S$ and $I(T^* + 1) > S$. This equation is not only simple to calculate, independent of a requirements function $F(t)$, but the approximation does something which Silver and Meals' approximation does not. Equation (6) states that at the optimal T^* the inventory costs approach, but are less than, the fixed cost of one order. If we try to insert one additional order at $T < T^*$, the total cost over T^* can only increase. This is because the inventory and order cost are equal, and the reduction in inventory cost cannot be greater than the added cost of an inserted order.

On the other hand, if no approximation is made, or if the Silver and Meal approximation is made, it is sometimes more economical to insert an extra order. Consider the following decreasing requirements function: $F(t) = A/t$. If we find T^* directly from Eq. (3), the result is $T^* = \infty$. Yet an additional order at some $T < T^*$ is more economical because the added order cost is less than the savings in inventory cost. That is, some T exists such that Eq. (7) holds true.

$$S < h(T-1) \sum_{T}^{T^*} F(t) \text{ for some } T. \tag{7}$$

Rather than try to find the economical upper limit to T^* as dictated by Eq. (7), it is much more convenient to solve for T^* from Eq. (6) which guarantees that T^* is not above the limit.

Table 3: Sample problem for single product with no constraints using method of this paper.

	TIME PERIOD (T)												
	1	2	3	4	5	6	7	8	9	10	11	12	13
REQUIREMENT	10	10	15	20	70	180	250	270	230	40	0	10	10
SOLUTION	55	0	0	0	70	180	250	270	290	0	0	0	0
C' (T)	−300	−70	−25	− 6	+18	▨	▨	▨	▨	▨	▨	▨	▨
I (T)	$ 0	$ 20	$ 80	$200	$760	▨	▨	▨	▨	▨	▨	▨	▨

Method would pull only 55 units in the first period. To pull the next 70 units would not be economical because the cost per period would be increased by C' (T) = $18.

Sample Problem

The sample problem shown here is from Silver and Meal. Assume a cost per order, S = $300, and a cost to carry inventory, h = $2, per unit per period. The first row in Table 3 shows the requirements by period. The second row shows the final solution over 13 periods resulting from repeated use of Eq. (5). The first order cycle is 4 periods long. That is, an order for 55 units is placed in period 1 to cover periods 1 through 4. In period 5 another order is made and 70 units ordered to cover that period only, etc. The next row down in the table shows the calculations, using Eq. (5), necessary to determine that the length of the first order cycle shall be 4 periods. Putting $T = 1$ into Eq. (5) we get $C'(1) = -300$ as shown in the table. For $T = 2$ we get -70, etc. As long as the computed value is negative there is a positive reduction in cost associated with combining that period's requirement into the order size. Finally at $T = 5$ the computed value is positive (+18) indicating we do not wish to include period 5 requirement in with the order in period 1. The solution for the first order cycle is therefore to pull the requirements of periods 1 through 4 into the order in period 1. The procedure is repeated to compute the length of the second order cycle, then the third, etc.

In the last row of the table is the computed inventory cost for the first order cycle. Since the solution is $T^* = 4$, $200 is the inventory cost of the 55 units to be ordered in period 1. This is the cost of holding 10 units for no periods, plus 10 units for 1 period, plus 15 units for 2 periods, plus 20 units for 3 periods. Notice that $200 is as close to the $300 setup cost as possible without going over. Thus Eq. (6) is also satisfied.

It may be of interest to compare the total cost of this solution with other methods. The solutions and costs shown here are those which would result when requirements are known, and assuming that the first 12 periods' requirements repeat. In this example, the cost of the method in this paper is therefore optimal or least cost. Using the method of computing cost devised in this paper [Eq. (1)], the following comparison is made:

This Method	$172 per period
Wagner-Whitin	$172 per period
Silver and Meal	$178 per period
Wilson Formula	$223 per period
No Lot Sizing	$275 per period

A study done at IBM supports these results [4] [5]. An algorithm basically described by Eq. (6), called the Part/Period Algorithm, was compared to five other algorithms including Wagner-Whitin. 102 sets of randomly generated demand data, having degrees of fluctuation ranging from 10% to 100% of the mean, were used—assuming both short and long planning horizons. The study found that the Part/Period Algorithm averaged within ½ of one percent of minimum cost, and was about 20% better than the Wilson formula. For those demands having 100% fluctuations, the Part/Period Algorithm was about 45% better than the Wilson formula. When the planning horizon was shorter than the two order cycles, Part/Period outperformed Wagner-Whitin by about 1.6%.

Capacity Constraint and Many Products

With a single product and no capacity constraint, the length of the first order cycle was found by increasing T until no more reduction in cost per period was possible, according to Eq. (5). The order size for period 1 was the summation of all requirements through $T = T^*$. However, with a capacity constraint in period 1, it may not be possible to increase T this far because total requirements pulled into period 1 may exceed the constraint. With many products we must decide how much of each to make.

The order sizes for period 1 are found by increasing T one period for that product which shows the greatest potential reduction in cost per period for each additional requirement this pulls into the order. This is repeated until the capacity constraint is violated or until no additional cost reduction is possible for any product. The reduction in cost per period is the negative of Eq. (5). For each product j, Eq. (8) gives the reduction in cost per period at Time T divided by the incremented quantity pulled into the order

$$U(T)_j = -C'(T)_j/F(T)_j = \{[S-I(T)]/T^2\}_j/F(T)_j \quad (8)$$

Equation (8) is used to decide for which product to increment T next. If no constraint exists then we would continue to increment T until $T = T^*$ and $U(T) = 0$ for each and every product. In other words, we would get the same answer as if we had treated each product independently, and according to Eq. (5). However, when constraints do exist, the procedure will minimize total cost per period for all products taken collectively.

The Algorithm

Step 1

A "matrix of requirements" is shown in Table 4. The rows are products, and the columns are time periods. Product #2 is the same as illustrated previously in this paper. Products which are available in inventory will not show any production requirements, and 0's will appear in the first columns. Appendix C explains how to modify the requirements matrix where constraints exist on individual order sizes.

Table 4: Initial requirements matrix.

PRODUCT (j) \ WEEK (T)	1	2	3	4	5	6	7	8	9	10	11	12
1	50	75	0	35	15	25	7	10	5	3	0	5
2	10	10	15	20	70	180	250	270	230	40	0	10
3	5	3	7	30	10	12	10	25	0	10	30	14
4	80	105	30	70	10	90	25	84	75	150	63	0
5	45	20	55	35	64	36	40	70	48	53	63	28
6	0	50	12	8	30	13	2	0	0	40	16	71
7	2	8	3	13	16	10	25	20	34	28	45	33
8	8	15	0	35	26	43	0	8	16	0	33	20
9	0	130	40	80	0	16	9	3	5	1	0	2
10	100	50	100	50	100	50	100	50	100	50	100	50

Step 2

Compute the incremental cost reductions per unit quantity added to the orders in the first period. Determine by using Eq. (8) the values $U(T)_j$ for each product j, and for each time period T. The computed values have been multiplied by 10 and placed in the upper right hand corner of the requirements matrix as shown in Table 5. Notice that for each product these values are generally decreasing with time, and that negative values need not be computed. Assuming a cost per order, $S = \$300$, and a holding cost per unit period, $h = \$2$, we may illustrate the computation for product #2 period 3:

$U(3)_2 = \{300 - 2\,[0 + 10 + 2\,(15)]\}/3^2/15 = 1.64$ and a value of 16 has been placed in Table 5.

Step 3

Determine total units to be ordered in the first period of the requirements matrix. It is often necessary to order in period 1 at least the period 1 requirements. This is because the penalty of missing a requirement can be relatively high. In any case, we probably would not want to order for future periods for one product while not ordering immediate requirements for another product. If this is the case, our action is to attempt to pull into the first period order, all requirements in period 1. In the sample problem of Table 5, capacity in period 1 is assumed to be 500 units while requirements are only 300 units, and there is no problem. If period 1 requirements had exceeded 500 we would not have been able to combine requirements, but would have

Table 5: Procedure for pulling requirements.

PRODUCT (j) \ WEEK (T)	1	2	3	4	5	6	7	UNCONSTRAINED ORDER SIZE	(X'S) ACTUAL ORDER SIZE
1	X $\boxed{250}$ 50	X $\boxed{60}$ 75	X $\boxed{\infty}$ 0	$\boxed{15}$ 35	$\boxed{4}$ 15	$\boxed{2+}$ 25	$\boxed{}$ 7	200	125
2	X $\boxed{300}$ 10	X $\boxed{70}$ 10	$\boxed{16}$ 15	$\boxed{3}$ 20	$\boxed{-3}$ 70	$\boxed{}$ 180	$\boxed{}$ 250	55	20
3	X $\boxed{400}$ 5	X $\boxed{90}$ 3	X $\boxed{65}$ 7	$\boxed{5}$ 30	$\boxed{2+}$ 10	$\boxed{}$ 12	$\boxed{}$ 10	55	15
4	X $\boxed{100}$ 80	$\boxed{2-}$ 105	$\boxed{}$ 30	$\boxed{}$ 70	$\boxed{}$ 10	$\boxed{}$ 90	$\boxed{}$ 25	185	80
5	X $\boxed{200}$ 45	X $\boxed{80}$ 20	$\boxed{20}$ 55	$\boxed{15}$ 35	$\boxed{5}$ 64	$\boxed{3}$ 36	$\boxed{1}$ 40	295	65
6	X $\boxed{\infty}$ 0	$\boxed{}$ 50	$\boxed{}$ 12	$\boxed{}$ 8	$\boxed{}$ 30	$\boxed{}$ 13	$\boxed{}$ 2	0	—
7	X $\boxed{500}$ 2	X $\boxed{100}$ 8	X $\boxed{95}$ 3	X $\boxed{55}$ 13	$\boxed{30}$ 16	$\boxed{10}$ 10	$\boxed{2}$ 25	52	26
8	X $\boxed{320}$ 8	$\boxed{35}$ 15	$\boxed{\infty}$ 0	$\boxed{3}$ 35	$\boxed{}$ 26	$\boxed{}$ 43	$\boxed{}$ 0	58	8
9	X $\boxed{\infty}$ 0	$\boxed{}$ 130	$\boxed{}$ 40	$\boxed{}$ 80	$\boxed{}$ 0	$\boxed{}$ 16	$\boxed{}$ 9	0	—
10	X $\boxed{150}$ 100	X $\boxed{40}$ 50	$\boxed{-1}$ 100	$\boxed{}$ 50	$\boxed{}$ 100	$\boxed{}$ 50	$\boxed{}$ 100	150	150
							TOTALS	1050	489
							CAPACITY	500	500

had to solution period 1 by some other means. In this case, however, all of period 1 requirements are assumed to be included in the orders.

Additional units are pulled into the order in chronological sequence according to decreasing values of $U(T)_j$. In the example, the next set of requirements pulled into an order, after all first period requirements, were the 8 units for product #7, period 2. The value of 100 in the upper right hand corner of the cell is clearly higher than any other cell not yet pulled without breaking sequence. The requirements for product #7, period 3, are pulled next, and the 3 units of product #3, period 2, after that. The procedure is followed until either capacity for period 1 is exceeded, or until there are no more requirements with positive values of $U(T)_j$ to be pulled in. In Table 5 an X is placed in the upper left hand corner of each cell to indicate that its requirements are to be pulled into the order. Referring to Table 5, in its present state, the next cell to be pulled would be the 15 units for product #8, period 2. However, if we add 15 units to the 489 already pulled, the capacity of 500 is exceeded. Therefore, we cannot pull the 15 units, and as there are no other cells with positive values and quantities less than 11 units, the procedure is now complete for period 1. The right hand column in Table 5 indicates the total amount to be ordered in period 1 for each product.

Step 4

Establish a new matrix of requirements with old period 2 as the new period 1. Notice that requirements previously pulled into period 1 will no longer exist in the new matrix. See Table 6.

Table 6: New requirements matrix after solutioning week 1.

		WEEK (T)											
		2	3	4	5	6	7	8	9	10	11	12	13
PRODUCT (j)	1	0	0	35	15	25	7	10	5	3	0	5	?
	2	0	15	20	70	180	250	270	230	40	0	10	?
	3	0	0	30	10	12	10	25	0	10	30	14	?
	4	105	30	70	10	90	25	84	75	150	63	0	?
	5	0	55	35	64	36	40	70	48	53	63	28	?
	6	50	12	8	30	13	2	0	0	40	16	71	?
	7	0	0	0	16	10	25	20	34	28	45	33	?
	8	15	0	35	26	43	0	8	16	0	33	20	?
	9	130	40	80	0	16	9	3	5	1	0	2	?
	10	0	100	50	100	50	100	50	100	50	100	50	?

Step 5

Calculate production for period 1 of new matrix by repeating steps 2 through 4.

Step 6

Repeat steps 2 through 5 until production has been determined for the desired number of periods. This final result represents an ordering plan. A possible 6-week ordering plan which may have resulted is shown in Table 7.

Table 7. Resultant ordering plan 6 weeks.

		WEEK (T)					
		1	2	3	4	5	6
PRODUCT (j)	1	125	0	0	100	0	0
	2	20	0	105	0	0	180
	3	15	0	0	92	0	0
	4	80	215	0	0	0	199
	5	65	0	90	0	140	0
	6	0	70	0	0	45	0
	7	26	0	0	0	133	0
	8	8	15	0	128	0	0
	9	0	170	0	116	0	0
	10	150	0	150	0	150	0
TOTAL		489	470	345	436	468	379

Implementation

The problem with implementing a lot sizing system is that we cannot start right out lot sizing all products without regard to capacity and production smoothing. If we do so, there will be a sudden bulge of production in the first period followed by a void in the second period. The algorithm presented here allows lot sizing to be implemented gradually. The solution for time period 1 will only combine requirements from future periods until the capacity limit is reached. It may be that in period 1 all products will have order sizes less than their full unconstrained economic lot size. After pulling requirements into period 1, there will be more excess capacity in period 2. More requirements can then be pulled into period 2. The solution for period 2 frees up even more excess capacity in period 3. Finally a period is reached when every product will be made at its full unconstrained lot size.

The problem of lot sizing goes beyond implementation. After implementation there will be fewer orders, and more pieces per order. Ideally these orders will remain staggered in time such that each order can be at its optimal unconstrained size. Realistically, however, conditions such as inventory costs, setup costs, demands, and capacity, are dynamic. Therefore, sudden excessive demand on production capacity is constantly a threat. The method of this paper continues to prevent production from exceeding capacity after implementation.

The subject of lot sizing cannot be separated completely from production smoothing. The lot sizing method presented here provides some degree of smoothing. As production in certain periods is not allowed to go above the capacity limit, production in other periods automatically remains high. The reader is asked to consider the effect on production smoothing if an artificial capacity limit were set at a level just above the average total production rate. Nevertheless, after use of the lot sizing algorithm it may be desirable to do additional production smoothing. A separate procedure can be developed which moves whole orders into earlier periods after lot sizing is complete. Such a procedure would not destroy the lot sizes already developed.

Acknowledgment

I would like to thank Messrs. Stephen Custer and John Mahoney of IBM East Fishkill for their efforts. They helped to refine the concepts involved, and assisted in verifying the validity and worth of the algorithm. Due to their efforts the algorithm is being programmed and implemented as part of a new production scheduling system servicing about 2000 different manufactured products.

Appendix A—Computing the Holding Cost

The holding cost h is assumed to include not only the cost of money and space, but also the risk cost of scrap due to obsolescence. If scrap-causing events are assumed to occur randomly, then one may compute:

$$h = k_1 \cdot [1 - \exp(-\lambda)] + k_2 + k_3 \cdot i \tag{A1}$$

where k_1 is the cost per unit scrapped, λ is the expected number of scrap causing events per period, k_2 is warehousing cost per unit, k_3 is the value of 1 unit, and i is the interest rate of tying up funds.

Appendix B—Wilson Formula

$$\text{Constant Order Size in } \$ = \sqrt{\frac{2 \cdot L \cdot A}{I}} \tag{B2}$$

where

I is holding rate expressed as a fraction

L is setup or fixed cost per order

A is yearly demand in \$ (cost)

Appendix C—Constraint on Individual Order Size

There may be an upper limit on the size of an individual order. If this is the case, an additional step is required prior to step 1. The purpose of this added step is to subtract from the starts requirements matrix, and from the capacity constraints, all requirements for which order sizing should not be done.

Suppose that in some period T the requirements are 180 and that a maximum order size is 50. There would be no point to starting 3 orders of 50 each in an earlier period, as this would only increase inventory without reducing the number of orders. Thus the 3 lots of 50 must be ordered in period T. The remaining 30, however, can be considered for combining with up to 20 requirements in earlier periods. Since we wish to consider only 30 for order sizing, we remove 150 from the requirements matrix, and also from the capacity constraint in period T before proceeding. An easy way to compute this is to use modular division: 180 modular 50 = 30. The procedure then proceeds step by step as described in the text except that, in addition, no single order is allowed to exceed the order size constraint. The removed requirement (150) may be added back into the resultant order size matrix after the procedure is complete.

References

[1] Wilson, R. H., "A Scientific Routine for Stock Control," *Harvard Business Review*, **XIII**, 1 pp. 116-128 (October 1934).

[2] Wagner, H. M. and Whitin, T. M., "Dynamic Version of the Economic Lot Sizing Model," *Management Science* (October 1958).

[3] Meal, Harlan C. and Silver, Edward A., "A Simple Modification of the EOQ for the Case of a Varying Demand Rate," *Production and Inventory Management*, (Fourth Quarter 1969).

[4] Matteis, J. J., "Part/Period Algorithm–A New Economic Lot Sizing Technique," Technical Report 17-220, IBM A.S.D.D., Yorktown Heights, New York (August 1967).

[5] Matteis, J. J., "An Economic Lot-Sizing Technique I, The Part-Period Algorithm," *IBM Systems Journal*, **7**, 1 (1968).

Mr. Peter S. Eisenhut is employed by IBM East Fishkill where he is responsible for developing advanced techniques for Planning and Industrial Engineering. He holds a BME degree from Cornell University and an MBA from the University of Rochester. Previously he was employed by Gleason Works, Rochester, New York, as a systems analyst. Mr. Eisenhut is a Director of the Mid-Hudson Chapter of AIIE.