Multi-periodic joint replenishment planning method for various all-unit discounts

Agathe Métaireau, Rabin Kumar Sahu, Simon Delecourt, Alexandre Gerussi, Manuel Davy

Multi-periodic joint replenishment planning method for various all-unit discounts

Agathe Métaireau[1,2], Rabin Sahu[1], Simon Delecourt[1], Alexandre Gerussi[1], Manuel Davy[1]
agathe.metaireau.etu@univ-lille.fr, rsahu@vekia.fr, sdelecourt@vekia.fr, agerussi@vekia.fr, mdavy@vekia.fr

[1]*Vekia, 143 rue d'Athènes, Lille, France*
[2]*Univ. Lille, CNRS, Centrale Lille, UMR 9189 - CRIStAL, France*

Keywords: Joint replenishment, Multi-period planning, Inventory control, Integer Linear Programming

Abstract: This paper aims at developing a multiperiodic joint replenishment optimization method to tackle all-unit discounts. In many industrial contexts, there exist multi-item constraints that can't be treated by single-item optimization. For example, suppliers can charge a fixed ordering cost or offer a discount above a given ordered quantity. To tackle such constraints, we set in place a model based on ordering blocks. This modeling enables to reduce the search-space by predetermining a given set of possible order quantities. The model is then solved using an Integer Linear Program that outputs the ordering plan for a given time horizon. This Integer Linear Program searches for the ordering block combination that gives the minimal cost. The cost function includes single-item costs like purchase or inventory costs as well as multi-item costs. The methodology was tested on twenty generated instances and compared with a single-period single-item replenishment engine. We show that our multiperiodic joint replenishment approach allows a reduction of the costs and an increase in the service level.

1 INTRODUCTION

A large part of the research conducted on inventory replenishment planning focuses on single-item control. In many references, the target is to find the best policy and its parameters. The final goal is to reach optimal or near-optimal inventory control for given situations and different objectives. There are however several industrial contexts where treating items separately is not the best strategy. In these contexts, there exist interactions between items. Fixed ordering cost or all-unit and incremental discounts are examples of this aspect. In such contexts, the problem cannot be reduced to several single-item sub-problems, because the constraints apply to the whole order. In this paper, we will only consider first-order interactions between the items, which can happen in various situations, in addition to the ones we mentioned above.

As this type of problem cannot be decomposed, there is a need to develop strategies to jointly replenish the items under control. Moreover, reaching all-unit or incremental discounts sometimes requires ordering more than the optimal quantity we compute for one coverage period, and considering several periods is a way to reach the discounts with relevant ordering quantities, which is an incentive to consider multi-periodic replenishment planning.

As a result, we will consider in this article a multi-item multi-period replenishment problem, with a fixed cost constraint and for various types of all-unit discounts. The multi-periodic joint replenishment problem has already been addressed in the literature for different contexts. Some authors calculate the Economic Order Quantity for this context, like [Das et al., 2000], who use geometric programming and gradient-based non-linear programming on capacitated systems and [Mousavi et al., 2013], who show the performance of a genetic algorithm with a mix of incremental and all-unit discount. [Mandal et al., 2011] use fuzzy techniques and genetic algorithms to address the optimal production control problem with the demand rate being stock-dependent. [Rezaei and Davoodi, 2008] use genetic algorithms in the context of defective items with supplier selection. [Gao et al., 2020] use dynamic programming to find a multi-raw material control policy under carbon emission constraints. [Mirzapour Al-e-Hashem and Rekik, 2014] developed a Mixed-Integer Linear Program (MILP) to solve an inventory routing problem with deterministic demand and transshipment allowed.

The multi-period joint replenishment problem is

NP-Hard ([Sahu, 2020]). Therefore, it is impossible to solve it exactly in a reasonable time, and it is necessary to solve it approximately. In this article, we tried to reduce the solution space of the problem we address by modeling. In this model, the orders we can place are represented by objects we call "ordering blocks", as defined in [Sahu, 2020]. The optimization aims to find the best combination of blocks for the considered horizon. It enabled us to formulate the problem we want to solve as an Integer Linear Program (ILP) and to solve it using a classic branch-and-bound solver. As there are multiple cases of all-unit discounts in industrial replenishment contexts, the ILP has been written to be generic and applicable to a large set of discount types.

The ILP allows an easy formulation of the problem. As free and commercial solvers already exist, there is no need to develop more complicated heuristics to solve ILPs. As a result, the method we offer is easily implementable in real multiperiodic joint-replenishment contexts. Moreover, the optimization tool we develop is intended to be a module assisting a single-item single-period replenishment optimization engine. Therefore, it is suitable to be integrated with already existing optimization tools.

The remainder of the paper is organized as follows. We will first describe the approach we developed and the notations we use in section 2. Then, we will present the ILP in section 3. We will next describe the experimental protocol we put in place to validate the model and present the numerical results in section 4. Finally, we will provide a conclusion in section 5.

2 DEVELOPED APPROACH AND NOTATIONS

In this section, we will first detail the problem we address and give some examples of application in industrial replenishment contexts. Then, we will explain the approach we put in place to model and solve it.

2.1 Details of the Problem and Use Cases

As we stated before, we focus on constraints concerning several items at the same time. It implies a need to design an optimization method that takes into account all the items simultaneously. We search to plan the replenishment for groups of items jointly, on several periods for a finite horizon, considering joint costs and discounts. In this paper, we consider only all-unit discounts. This means the discount is applied on all the items ordered once the breakdown is reached. We apply a direct grouping strategy, as defined in [Bastos et al., 2017], and consider only groups of items that have the same cycle time.

There are two major categories of multi-item constraints we consider: fixed ordering cost and all-unit discount. They are defined as follows.

Fixed ordering cost : It is a cost applied to the whole order once we order something (*i.e.* when the total quantity ordered is strictly superior to 0). For example, a supplier can add a transport cost to the ordering cost, and this cost is only applied if something is ordered. This cost is also called in the literature "*Major Ordering Cost*".

All-unit discount : It is a discount applied on all the items ordered (*i.e.* on the total ordering cost) once a given breakdown is reached. There can be one or several breakdowns with corresponding discounts. The breakdowns can also concern different values, such as the total ordering cost or the total quantity ordered, but also the total weight or volume of the order for example. A practical example of the all-unit discount is the Franco, where the supplier charges a constant penalty cost if the total ordering cost is lower than a given breakdown.

Figures 1 and 2 are graphical examples of the two notions explained above.

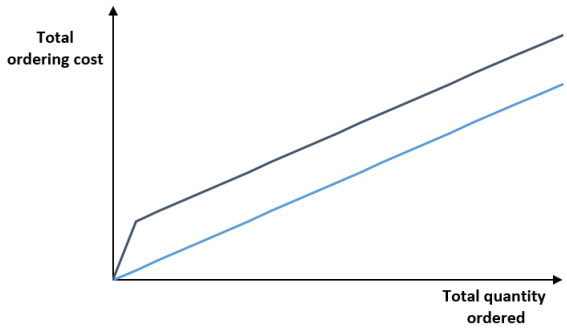

Figure 1: Representation of the cost function in case of a fixed ordering cost. The dark blue curve is the case without fixed ordering cost.

2.2 Modeling Approach and Definitions

As the problem is NP-Hard, we can't expect to find the optimal solution to large-scale instances in a reasonable time. Therefore, we have to define a process to find good-quality solutions in a suitable delay. In this article, we choose to reduce the search

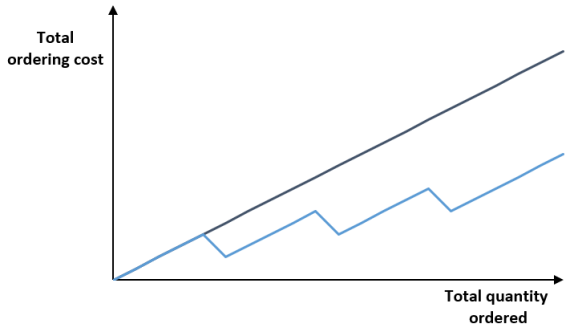

Figure 2: Representation of the cost function for an all-unit discount with three breakdowns and three corresponding fixed discounts. The dark blue curve is the case without any discount.

space by modeling. First, we constrain the time steps at which an order can be placed. It seems to be a reasonable choice, as the ordering dates are frequently constrained in real-life replenishment contexts.

We define the notion of coverage period as a time interval $[t_1, t_2]$ for which an order can be placed at t_1 and no order can be placed until t_2. In other words, this is the smallest possible interval between two orders.

We also choose to constrain the quantities that can be ordered. At the beginning of every coverage period, the decision-maker can only choose among a set of possible ordering quantities. These quantities are the optimal ordering quantities for t coverage periods, with t going from 1 to the end of the planning horizon. They are calculated before the multi-periodic joint replenishment optimization.

As a result, we represent the possible orders as an object called "block". Every block contains several attributes. First, it has a length, which is the number of coverage periods it encompasses. It has then a quantity to order, which is the precomputed optimal quantity for the block's length, and a cost, which is the total cost incurred for ordering the optimal quantity. The block's cost involves the purchase cost, the holding cost, the lost-sale or backorder cost, and any relevant cost for the problem under study. A block can also contain another attribute named value, which represents the value compared to the discount breakdown. This value can be the purchase cost, the volume, the weight, *etc*. Figure 3 represents all the possible blocks for one item and a time horizon of three coverage periods.

2.3 Notations

In this section, we describe the notations we use in the remainder of the article. All the notations we will

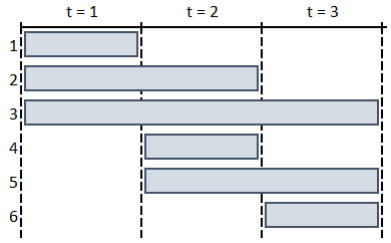

Figure 3: Possible blocks for one item and a time horizon of three coverage periods.

use are listed below. The subscripts t refer to the time steps, n to the items, and i to the cost breakdowns.

T - Time horizon

N - Size of the item set

I - Number of cost breakdowns

K - Fixed ordering cost

$B_{nt_1t_2}$ - Block corresponding to the order of item n from coverage period starting at t_1 and ending at t_2

$q_{nt_1t_2}$ - Quantity to order for the block $B_{nt_1t_2}$

$C_{nt_1t_2}$ - Cost incurred by the block $B_{nt_1t_2}$

$V_{nt_1t_2}$ - Value of the block $B_{nt_1t_2}$ to be compared to the breakdowns

l_i - Breakdown i

f_i - Fixed cost or discount for a total order value under the breakdown i

u_i - Cost or discount per unit for a total order value under the breakdown i

2.4 Solution Approach

The goal of the multi-periodic joint replenishment optimization method we develop in this article is to find what quantity of each item to order at every time step of the planning horizon while minimizing the total cost. This total cost, which we call TC in this section, includes the block's cost and the costs and discounts incurred by the multi-item constraints.

Let P_{NT} be the set of possible block combinations for an item set of size N and a planning horizon T. As an example, a representation of the set P_{22} is provided in Figure 4.

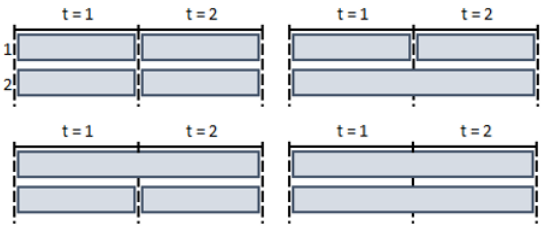

Figure 4: Representation of the set of possible block combination for two items and a planning horizon of two periods

Given N and T, each possible block combination

p in P_{NT} has a total cost $TC(p)$, which is the sum of the costs of the blocks selected in p and the corresponding multi-item costs and discount. To find the best replenishment planning in this context, we are looking for the block combination p^* with the minimum cost, as described in the equation 1.

$$p^* = \arg\min_{p \in P_{NT}} TC(p) \tag{1}$$

In the next section, we will describe the methodology we use to find p^*.

3 OPTIMIZATION METHODOLOGY

An intuitive solution to solve the problem we described above would be to generate every possible block combination, compute their respective total cost, and choose the one with the lowest total cost. However, the number of possible block combinations for a time horizon T and an item set of size N is $2^{N\times T-1}$. It means the number of possible block combinations quickly increases with the growth of both T and N. For example, if $T = 5$ and $N = 5$, there are 1 048 576 possibilities, which illustrates how the brute force approach rapidly becomes intractable.

To tackle this, we chose to solve the problem by using an Integer Linear Program (ILP). This optimization method has the advantage of being easy to formulate, implement, and solve using commercial or free solvers.

We will present the formulation of the ILP in the next sections. Section 3.1 lists the data used in the ILP, Section 3.2 details its variables. Section 3.3 is dedicated to the objective. Section 3.4 lists all the constraints and provide some explanation of them. Section 3.5 details the domains of the variables.

3.1 Data

T - Time horizon
N - Set of items
I - Number of breakdowns
K - Fixed ordering cost
M - Suitably large number
$\forall n \in N, \forall t_1 \in [\![1,T]\!], \forall t_2 \in [\![1,T]\!], C_{nt_1t_2}$ - Cost of the block $B_{nt_1t_2}$
$\forall n \in N, \forall t_1 \in [\![1,T]\!], \forall t_2 \in [\![1,T]\!], V_{nt_1t_2}$ - Value of the block $B_{nt_1t_2}$
$\forall i \in [\![1,I]\!], l_i$ - i-th breakdown
$\forall i \in [\![1,I+1]\!], f_i$ - Fixed cost or discount corresponding to a value ordered over l_{i-1} and under l_i
$\forall i \in [\![1,I+1]\!], u_i$ - Cost or discount per unit corresponding to a value ordered over l_{i-1} and under l_i

3.2 Variables

$\forall n \in N, \forall t_1 \in [\![1,T]\!], \forall t_2 \in [\![1,T]\!],$

$$x_{nt_1t_2} = \begin{cases} 1 & \textit{if the block } B_{nt_1t_2} \textit{ is selected} \\ 0 & \textit{otherwise} \end{cases}$$

$\forall t \in [\![1,T]\!],$

$$\eta_{t1} = \begin{cases} 1 & \textit{if the value ordered is } 0 \\ 0 & \textit{otherwise} \end{cases}$$

$\forall t \in [\![1,T]\!],$

$$\eta_{t2} = \begin{cases} 1 & \textit{if the value ordered is} \\ & \textit{strictly between } 0 \textit{ and } l_2 \\ 0 & \textit{otherwise} \end{cases}$$

$\forall t \in [\![1,T]\!], \forall i \in [\![3,I]\!]$

$$\eta_{ti} = \begin{cases} 1 & \textit{if the value ordered is between} \\ & l_{i-1} \textit{ and } l_i \textit{ excluded} \\ 0 & \textit{otherwise} \end{cases}$$

$\forall t \in [\![1,T]\!],$

$$\eta_{tI+1} = \begin{cases} 1 & \textit{if the value ordered is greater} \\ & \textit{than } l_I \\ 0 & \textit{otherwise} \end{cases}$$

$\forall t \in [\![1,T]\!], \forall i \in [\![1,I+1]\!]$

$$y_{ti} = \begin{cases} \textit{Value ordered at } t & \textit{if } \eta_{ti} = 1 \\ 0 & \textit{otherwise} \end{cases}$$

3.3 Objective

$$\min \sum_{n\in N}\sum_{t_1=1}^{T}\sum_{t_2=1}^{T} C_{nt_1t_2}x_{nt_1t_2} + \sum_{t=1}^{T} K(1-\eta_{t1}) + \sum_{t_1=1}^{T}\sum_{t_2=1}^{T}\sum_{i=1}^{I}(u_i y_{ti} + f_i \eta_{ti})$$

3.4 Constraints

$\forall n \in N, \forall t_1 \in [\![1,T]\!], \forall t_2 \in [\![1,T]\!],$

$$t_1 x_{nt_1t_2} \leqslant t_2 x_{nt_1t_2} \tag{2}$$

$$\forall n \in N, \sum_{t_1=1}^{T}\sum_{t_2=1}^{T}(t_2 - t_1 + 1)x_{nt_1t_2} = T \tag{3}$$

$$\forall n \in N, \sum_{t=1}^{T} x_{ntT} = 1 \tag{4}$$

$$\forall n \in N, \forall t \in [\![1, T-1]\!], \sum_{t_1=1}^{T} x_{nt_1t} \leqslant \sum_{t_2=1}^{T} x_{nt+1t_2} \quad (5)$$

$$\forall t \in [\![1, T]\!], M(1-\eta_{t1}) \geqslant \sum_{n \in N} \sum_{t_2=1}^{T} V_{ntt_2} x_{ntt_2} \quad (6)$$

$$\forall t \in [\![1, T]\!], (1-\eta_{t1}) \leqslant \sum_{n \in N} \sum_{t_2=1}^{T} V_{ntt_2} x_{ntt_2} \quad (7)$$

$$\forall t \in [\![1, T]\!], \forall i \in [\![2, I+1]\!],$$
$$l_{i-1}\eta_{ti} \leqslant \sum_{n \in N} \sum_{t_1=1}^{T} V_{ntt_2} w_{ntt_2} \quad (8)$$

$$\forall t \in [\![1, T]\!], \forall i \in [\![2, I]\!],$$
$$M(1-\eta_{ti}) \geqslant \sum_{n \in N} \sum_{t_2=1}^{T} V_{ntt_2} x_{ntt_2} + \varepsilon \quad (9)$$

$$\forall t \in [\![1, T]\!], \sum_{i=1}^{I+1} \eta_{ti} = 1 \quad (10)$$

$$\forall t \in [\![1, T]\!], \forall i \in [\![1, I+1]\!],$$
$$y_{ti} \geqslant \sum_{n \in N} \sum_{t_2=1}^{T} (V_{ntt_2} x_{ntt_2}) - M(1-\eta_{ti}) \quad (11)$$

$$\forall t \in [\![1, T]\!], \forall i \in [\![1, I+1]\!],$$
$$y_{ti} \leqslant \sum_{n \in N} \sum_{t_2=1}^{T} V_{ntt_2} x_{ntt_2} \quad (12)$$
$$\forall t \in [\![1, T]\!], \forall i \in [\![1, I+1]\!], y_{ti} \leqslant M\eta_{ti} \quad (13)$$

Constraint 2 ensures that we select only blocks for which the starting coverage period is lower than or equal to the ending coverage period. Constraint 3 ensures the entire planning horizon is covered. Constraints 4 and 5 ensure that the blocks we select do not overlap. Constraints 6, 7, 8, and 9 rule the value of η_{ti} according to its definition. If the total value ordered can be lower than 1, it is necessary to add a "$+\varepsilon$" at the end of the right part of constraint 7 to ensure its good functioning. The "$+\varepsilon$" at the end of the left part of constraint 9 allows having the behavior of a strict inequality. Constraint 10 ensures only one η_{ti} is selected each period. Constraints 11, 12, and 13 rule the value of y_{ti} according to its definition.

3.5 Domains

$$\forall n \in N, \forall t_1 \in [\![1, T]\!], \forall t_2 \in [\![1, T]\!], x_{nt_1t_2} \in \{0, 1\}$$

$$\forall t \in [\![1, T]\!], \forall i \in [\![1, I]\!], \eta_{ti} \in \{0, 1\}$$

$$\forall t \in [\![1, T]\!], \forall i \in [\![1, I]\!], y_{ti} \in \mathbb{N}$$

The domain of y_{ti} is consistent with the unit of V. We consider the case where V only takes integer values, but if V takes continuous values, the Integer Linear Program becomes a Mixed Integer Linear Program (MILP).

4 EXPERIMENTATIONS AND COMPUTATIONAL RESULTS

To evaluate the interests of multiperiodic joint replenishment planning and the performance of the method we developed, we chose to compare it with a single-period single-item optimization method. We used a single-period single-item ordering engine as a baseline for our experimentations. This engine uses a sample-based optimization approach for stochastic replenishment planning. Given a set ov demand samples, this engine aims to find the optimal quantity to order on a fixed coverage period. It will minimize the so-called "coverage period cost", which is the expected cost on the whole period, and includes the holding cost, the lost-sale cost, and an potential single-item fixed cost. The optimal quantity is found by using an enumerativesearch heuristic, described in [Sahu, 2020].

Both order engines are compared on the total cost performance criterion, which includes the purchase cost, the inventory cost, the shortage cost, and the multi-item cost. In the experiments, the multi-item constraint is a Franco penalty, as described in the section 2.1, so the multi-item cost includes only eventual Franco penalties.

To conduct experiments, we generated twenty different instances, each containing ten items. We assume perfect forecast, *i.e.* we assume that we perfectly know the demand distribution of the items. In every instance, all the demand distributions of the items are stationary Poisson distribution, and we assume all the unmet demand is lost. The parameters of the instances we used are detailed in the tables 1 and 2. The Integer Linear Program has been implemented

Instance n°	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8	λ_9	λ_{10}
1	10	10	10	10	10	10	10	10	10	10
2	13	25	14	15	41	12	20	34	19	34
3	39	44	13	21	15	47	31	14	29	8
4	45	39	20	11	11	9	9	10	42	10
5	24	30	14	34	42	35	24	35	40	46
6	18	17	14	32	22	23	7	11	11	19
7	47	33	50	25	48	38	16	11	18	33
8	39	47	41	26	31	28	13	32	47	36
9	30	6	17	18	34	50	46	48	50	10
10	36	26	14	18	48	11	44	21	15	16
11	24	41	42	37	22	26	45	7	42	49
12	50	46	48	43	16	13	46	6	38	42
13	20	48	13	20	41	48	15	36	6	48
14	21	34	18	47	29	31	44	44	34	44
15	38	44	22	25	38	42	9	49	5	30
16	22	11	13	36	15	20	35	43	31	7
17	37	38	12	18	45	49	47	30	46	31
18	44	15	18	8	45	30	10	21	36	41
19	19	31	42	40	37	50	5	16	23	22
20	11	43	43	19	19	46	25	7	7	31

Table 1: Details of the instances used for the experiments. λ_i represents the mean of the Poisson distribution for the demand of the i-th item of the instance. The global parameters are the purchase cost $p = 5$, storage cost $h = 0.1$ and lost sale cost $w = 10$.

Instance n°	Franco threshold	Franco penalty
1	800	276.35
2	2838	654
3	2250	753
4	2250	581.25
5	2150	939
6	1700	500
7	2400	920.5
8	2943	943
9	2150	899.5
10	2250	711
11	2889	963
12	2150	1019.95
13	2350	867
14	2050	999
15	2650	878.7
16	2250	668
17	3078	1025.95
18	1800	766
19	2484	828
20	1550	722.45

Table 2: Franco parameters of the instances.

using the PuLP library in Python, and solved using the COIN-OR Branch and Cut (CBC) solver.

To compare both replenishment processes, we used the following experimental protocol.

1. Generate demand samples for every item of the instance and every period of the planning horizon.
2. Given initial inventory levels, launch the single-item single-period order engine. Simulate the order and demand process and update the inventory levels.
3. Record the costs associated with the simulated period.
4. Repeat steps 1. and 2. on the whole planning horizon.
5. Compute the total cost for the single-item single-period ordering simulation.
6. Given the same initial inventory levels, launch the multi-period multi-item order engine. Record the replenishment planning.
7. Simulate the order and demand process using the same demand samples on the entire planning horizon and record the costs associated with every simulated period.
8. Compute the total cost for the multi-period multi-item ordering simulation.

This protocol was used on the twenty instances described above. It allowed us to compare the different costs incurred by both optimization methods, and

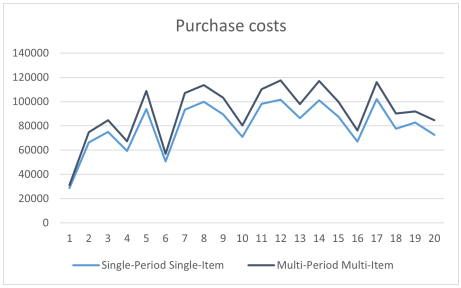

Figure 5: Comparison of the purchase costs given by the single-period single-item and multi-period multi-item approaches on all the instances.

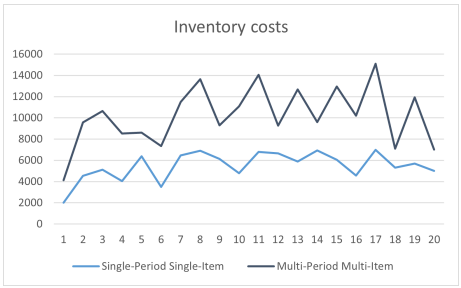

Figure 6: Comparison of the inventory costs given by the single-period single-item and multi-period multi-item approaches on all the instances.

conclude on the relevancy of the multiperiodic joint replenishment planning. The comparison of the different costs (purchase cost, inventory cost, shortage cost, and multi-item cost) is shown on the figures 5 to 8. The averaged savings are displayed in the table 3.

The values displayed in table 3 are defined as follows. For a given cost type, let C^j_{SPSI} and C^j_{MPMI} be respectively the total cost incurred by the single-period single-item approach for the instance j and the total cost incurred by the multi-period multi-item approach on the same instance. Then, let $S^j = TC^j_{SPSI} - C^j_{MPMI}$ be the total savings for instance j and $S^j_{\%} = \frac{TS^j}{C^j_{SPSI}}$. Let $\overline{S}$ and $\overline{S_{\%}}$ be their respective averaged values on all the instances.

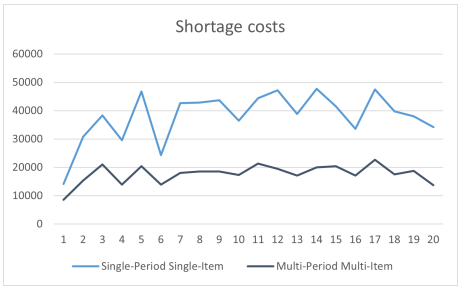

Figure 7: Comparison of the lost sale costs given by the single-period single-item and multi-period multi-item approaches on all the instances.

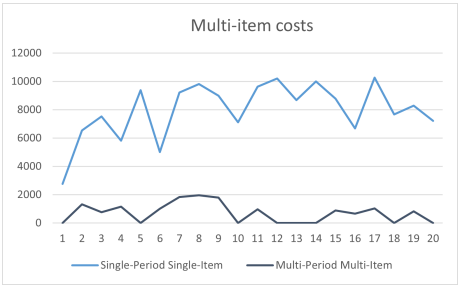

Figure 8: Comparison of the Franco costs given by the single-period single-item and multi-period multi-item approaches on all the instances.

Cost type	$\overline{S}$	$\overline{S_{\%}}$
Purchase	-11255.5	-13.76%
Inventory	-4725.38	-88.83%
Shortage	20427	52.72%
Multi-item	7267.32	91%
Total	11714.44	8.70%

Table 3: Averaged savings obtained with the multi-period multi-item approach compared to the single-period single-item approach.

Regarding the cost comparison, we obtained an overall cost reduction of 8.70% by using the multi-period multi-item approach. We also observe higher purchase and inventory costs, which can be explained by the fact that this order engine tends to order more than the single-period single-item one. The cost reduction comes from the lost sale cost and the multi-item cost. As a result, the proposed multi-period multi-item order engine allows to optimize the total cost regarding the multi-item constraints, as it was designed for, but it also allows to reduce lost sales, and therefore to increase the service level.

5 CONCLUSION AND FUTURE RESEARCH

In this paper, we present a method to optimize inventory replenishment by taking into account multi-item constraints. We develop a multi-period multi-item optimization tool to answer this problem. As the multi-periodic joint replenishment planning is an NP-Hard problem, we approximate the problem by modeling, to obtain good solutions in a reasonable time. To reduce the search-space, we introduced the concept of ordering blocks. This approximation seems to be relevant because it is in line with industrial ordering processes, and still adaptable to various situations. We wrote an Integer Linear Program to find the optimal

combination of blocks by minimizing the sum of the single-item costs (purchase, inventory, and shortage costs) and the multi-item costs (fixed ordering cost, Franco costs, discounts). This Integer Linear Program has been developed in the purpose to be adaptable to various all-unit discounts.

This multi-period multi-item approach was compared with a single-period single-item order engine. Using twenty generated instances, the performance of both methods was evaluated based on the costs obtained by simulation using a sample of the demand. We chose to restrict our study to stationary Poisson demand and to a single Franco constraint, but the model allows to treat various demand types and multi-item constraints. We showed that the multi-period multi-item order engine we developed incurred a reduction of the total cost on all the instances, especially on the multi-item costs and on the shortage cost. Therefore, this method allows better management of multi-item constraints and discounts and an increase in the service level.

For further research, the model can be tested with different discount types, like a fixed ordering cost or a fixed discount, as we only tested with a Franco here. It would also be interesting to evaluate the performances of the method in case of the imperfect forecast, to see how the demand uncertainty impacts the relevancy of this approach. The model can also be extended or adapted to tackle incremental discounts, which were excluded in this research.

References

[Bastos et al., 2017] Bastos, L. d. S. L., Mendes, M. L., Nunes, D. R. d. L., Melo, A. C. S., and Carneiro, M. P. (2017). A systematic literature review on the joint replenishment problem solutions: 2006-2015. *Production*, 27(2008):2006–2015.

[Das et al., 2000] Das, K., Roy, T. K., and Maiti, M. (2000). Multi-item inventory model with quantity-dependent inventory costs and demand-dependent unit cost under imprecise objective and restrictions: A geometric programming approach. *Production Planning and Control*, 11(8):781–788.

[Gao et al., 2020] Gao, X., Chen, S., Tang, H., and Zhang, H. (2020). Study of optimal order policy for a multi-period multi-raw material inventory management problem under carbon emission constraint. *Computers and Industrial Engineering*, 148(51705384):106693.

[Mandal et al., 2011] Mandal, S., Maity, A. K., Maity, K., Mondal, S., and Maiti, M. (2011). Multi-item multi-period optimal production problem with variable preparation time in fuzzy stochastic environment. *Applied Mathematical Modelling*, 35(9):4341–4353.

[Mirzapour Al-e-Hashem and Rekik, 2014] Mirzapour Al-e-Hashem, S. M. and Rekik, Y. (2014). Multi-product multi-period Inventory Routing Problem with a transshipment option: A green approach. *International Journal of Production Economics*, 157(1):80–88.

[Mousavi et al., 2013] Mousavi, S. M., Hajipour, V., Niaki, S. T. A., and Alikar, N. (2013). Optimizing multi-item multi-period inventory control system with discounted cash flow and inflation: Two calibrated meta-heuristic algorithms. *Applied Mathematical Modelling*, 37(4):2241–2256.

[Rezaei and Davoodi, 2008] Rezaei, J. and Davoodi, M. (2008). A deterministic, multi-item inventory model with supplier selection and imperfect quality. *Applied Mathematical Modelling*, 32(10):2106–2116.

[Sahu, 2020] Sahu, R. K. (2020). *General framework and optimization methods for stochastic replenishment planning in industrial contexts*. Theses, Université de Lille.