Robust inventory theory with perishable products

Marcio Costa Santos, Agostinho Agra, Michael Poss

Noname manuscript No.
(will be inserted by the editor)

Robust inventory theory with perishable products

Marcio Costa Santos · Agostinho Agra · Michael Poss

Received: date / Accepted: date

Abstract We consider a robust inventory problem where products are perishable with a given shelf life and demands are assumed uncertain and can take any value in a given polytope. Interestingly, considering uncertain demands leads to part of the production being spoiled, a phenomenon that does not appear in the deterministic context. Based on a deterministic model we propose a robust model where the production decisions are first-stage variables and the inventory levels and the spoiled production are recourse variables that can be adjusted to the demand scenario following a FIFO policy. To handle the non-anticipativity constraints related to the FIFO policy, we propose a non-linear reformulation for the robust problem, which is then linearized using classical techniques. We propose a row-and-column generation algorithm to solve the reformulated model to optimality using a decomposition algorithm. Computational tests show that the decomposition approach can solve a set of instances representing different practical situations within reasonable amount of time. Moreover, the robust solutions obtained ensure low losses of production when the worst-case scenarios are materialized.

Keywords Lot-sizing · Integer programming · Robust optimization · Row-and-column generation algorithms

1 Introduction

Dealing with uncertainty is very important when solving practical lot-sizing problems wherein production decisions need to be taken before the real demands are revealed. This issue is even more important when products are perishable because significant costs can be originated from lost production due to an overestimation of demands. Specifically, overestimating the demands increases the holding costs and the cost for the lost production since, typically, an item that reaches its shelf life will be either lost or sold at a residual value. Even worse, underestimating the demands leads to costs from supplying demands with delay which may include poor client satisfaction penalties and contractual costs. To overcome the limitation of these deterministic models, we consider in this paper a robust lot-sizing problem with recourse where the products have a fixed shelf-life. The demand of each period can be fulfilled by production in that period, from stock resulting from production in earlier periods within the shelf-life, or backlogged. The quantities to produce need to be decided in the beginning of the time horizon and the stock, the backlog and the lost demand are adjusted to the scenario.

M. C. Santos
Departamento de computação, IME, Universidade Federal da Bahia, Brazil
E-mail: mcs.marcio@gmail.com

A. Agra
CIDMA, Department of Mathematics, University of Aveiro, 3810-193 Aveiro, Portugal
E-mail: aagra@ua.pt

M. Poss
UMR CNRS 5506 LIRMM, Université de Montpellier 2, 161 rue Ada, 34392 Montpellier Cedex 5, France
E-mail: michael.poss@lirmm.fr

Robust optimization [6] has arisen in the past decades as an efficient model to incorporate demand uncertainty into the lot-sizing problems. The later assumes demand vectors can take any value in a given uncertainty set. In that case, different models arise, depending on whether the decision variables (e.g. production, stock, ...) can be adjusted to past history. One of the simplest models considered in the literature [8] supposes that productions are fixed before the planning horizon starts, while the stock variables can be adjusted to the actual value, which is equal to the total production minus the demand of the scenario. Although the productions are fixed, the stock variables must be modeled through adjustable variables, leading to a difficult multi-stage optimization problem. The work of [8] conservatively approximates the adjustable problem by a static one, yielding an easy optimization problem.

The original conservative approximation from [8] has been improved along two complementary lines of research. On the one hand, generic heuristic have been proposed, often based on decision rules [5] or dynamic partitions of the uncertainty sets [7,27]. The quality of these heuristic has then been assessed by sampling the uncertainty set [7] or using lower bounds based on duality [22] or perfect information [28]. An alternative line of research has been to solve these problems exactly [3,10,30]. The bottom line of these exact approaches lies in generating a subset of the elements of the uncertainty set. Then, these approaches iterate between a relaxed master problem, where the uncertainty set is now replaced by the finite set generated so far, and adversary separation problems that identify new uncertainty vectors to consider, using row-and-column generation algorithms. These approaches are typically suited for two-stage robust optimization with real recourse. This being said, some lot-sizing problems can be seen as two-stage problems, for instance when all production decisions are taken prior to knowing the demand. Thus, exact two-stage approaches have also been applied to solve specific lot-sizing problems [1,9], using specific types of uncertainty sets. We also mention the alternative exact approach proposed by [14], which can solve exactly certain lot-sizing problems by reformulating the later as *static* robust optimization problems, involving exponentially many constraints. We refer to [15,13,29] for comprehensive surveys on the developments of adjustable robust optimization.

In this paper, we consider an extension of the above lot-sizing problem by considering perishable products. The latter have a shelf-life that is typically smaller than the time horizon, after which they must be spilled. The subtlety with perishable products is that they are typically handled through FIFO policies, which means that oldest products are used first to attend the demand. This is an important consideration to take into account when deriving an optimization model, which has been analyzed in several papers (e.g. [20,23]).

While, to the best of our knowledge, the problem considered here has not been studied before, lot-sizing problems and models considering perishable products have been considered for decades [24]. For reviews of publications until 2011, see [4,25]. A more recent overview covering the years 2012 to 2015 is given by Janssen et. al. [20]. For overviews on managing perishability in production-distribution and supply chain planning see [2,26]. Recently, several studied have been conducted on inventory management of perishable products [11,16–18,21]. Among these references, stochastic models were considered in [16, 17] to handle uncertainty. However, no robust model of lot-sizing with perishable products has been considered so far, which is the gap this works intends to fill.

Our contributions The purpose of this paper is to provide an *exact* solution approach to the robust lot-sizing problem with perishable products, in line with the row-and-column generation algorithm from [1]. To do so, we first show how the problem can be reformulated without using any production variables. The resulting model resembles the static model for robust lot-sizing problems used in [1], however involving a non-convexity not present in [1]. Hence, we show how the latter problem can be solved exactly through a row-and-column generation algorithm, that generates elements of the uncertainty set on the fly. In particular, the non-convexity is handled using a classical big-M linearization. Our approach is assessed numerically on instances inspired by the scientific literature on robust lot-sizing, using a budgeted uncertainty set.

Structure of the paper In Section 2 we present the nominal and the robust models, and detail the impact of the FIFO policy on the models. We present our reformulation in Section 3. Section 4 describes the general row-column generation solution procedure. The computational experiments are reported in Section 5. Finally, the concluding remarks are given in Section 6.

2 Problem Description

2.1 The nominal problem

We are given a finite planning horizon $H = \{1, \dots, n\}$ together with a holding cost h_i, a backlogging cost p_i, a production cost c_i, a producing capacity C_i, and a spoiling cost q_i, for each period $i \in H$. We assume all these parameters are positive.

The product has a shelf life of m periods meaning that a product produced in time period i can be used to fulfill demand until period $i+m$, otherwise it is spoiled after period $i+m$. Client demands can also be fulfilled by backlogging. In this context, one wants to fulfill the client demands d_i for each period i by producing at that period, by stock or by backlogging, while respecting the production limits in each period.

In order to model a lot-sizing problem with perishable products it is necessary to keep track of the age of the inventories, which cannot be accomplished directly with basic lot-sizing formulations that only keep track of the total amount of stock at the end of each time-period. Different approaches have been used to handle this, for instance, in [12] a discretization of the age of the inventories is considered. Here we adapt to the inventory problem with perishable products the well-known facility location formulation for lot-sizing problem, because the variables used in this formulation keep track of the time the product is produced and consumed. Related formulations have been used for perishable products, see [19]. We define next the decision variables used in our formulation. Variables y_{ij}, for $i, j \in H$, represent the amount produced in period i to fulfill the demand of period j. We have $y_{ij} = 0$ for every $j > i+m$. Additionally, we consider variables $y_{n+1,j}$ that represent the amount of demand of period j that is not fulfilled within the time horizon, and variables $y_{i,n+1}$ that represent the amount of production in time period i that is not used within the time horizon and that is not yet spoiled. Variables s_i represent the amount stored from period i to period $i+1$, variables r_i represent the amount backlogged from period i to period $i+1$, variables π_i that represent the amount of production that is spoiled at time period i (amount produced in time period $i-m$ and not used until period i, period i included) and variables x_i represents all the amount produced at time period i. The nominal problem, denoted by $\mathcal{LT}$, can be modeled as follows.

$$\min \sum_{i \in H} (c_i x_i + h_i s_i + p_i r_i + q_i \pi_i) \tag{1}$$

$$\text{s.t. } s_i = \sum_{j=1}^{i} \sum_{k=i+1}^{\min(i+m,n)} y_{jk} + \sum_{j=i+1}^{\min(i+m,n)} \pi_j, \qquad \forall i \in H \tag{2}$$

$$r_i = \sum_{j=i+1}^{n+1} \sum_{k=1}^{i} y_{jk}, \qquad \forall i \in H \tag{3}$$

$$x_i = \sum_{j=1}^{i+m} y_{ij} + \pi_{i+m}, \qquad \forall i \in \{1, \dots, n-m\} \tag{4}$$

$$x_i = \sum_{j=1}^{n+1} y_{ij}, \qquad \forall i \in \{n-m+1, \dots, n\} \tag{5}$$

$$\sum_{i=\max(1,j-m)}^{n+1} y_{ij} = d_j, \qquad \forall j \in H \tag{6}$$

$$x \in X \tag{7}$$

$$x, y, \pi, r, s \geq 0 \tag{8}$$

The aim is to minimize the total cost. Constraints (7) represent the limitations imposed on the amount produced in each period. Unless stated otherwise we assume

$$X = \left\{ x \in \mathbb{R}^n_+ : x_i \leq C_i,\ \forall i \in H \right\}. \tag{9}$$

Equations (2) define the amount stored after all demands of period i have been attended. This amount considers the quantity produced in the previous periods to satisfy the demand in periods $i+1$ to $i+m$ plus the spoiled production. Equations (3) define the amount backlogged from time period i to time period $i+1$. Equations (4) and (5) establish that the amount produced in period i is used either to fulfill demand in periods 1 to $i+m$, or is lost. Equations (6) state that the demand in period j must be satisfied from production from periods $j-m$ to n or it is not satisfied during the time horizon (case $y_{n+1,j} > 0$). Constraints (8) are non-negativity constraints. Since the spoiling costs are positive, spoiling never occurs in an optimal solution to $\mathcal{LT}$.

Lemma 1 *Any optimal solution to* (1)–(8) *satisfies* $\pi = 0$.

The situation is however more subtle in the robust case where spoiling may happen, as we explain in the next subsection.

2.2 The robust problem

We consider in this paper the robust counterpart of problem $\mathcal{LT}$, where the demands are uncertain and belong to a known uncertainty set. Specifically, we assume that the client demands are affine functions $d_i(\xi) = \bar{d}_i + \hat{d}_i\xi$ of the elements $\xi \in \mathbb{R}^n$ belonging to a given *uncertainty polytope* Ξ, which we assume to be full-dimensional. We recall that $\bar{d}_i$ and $\hat{d}_i$ are numbers that represents the expected value of the client demand and their deviations, respectively. The correlation among the clients demands is modeled by set Ξ, which typically contains 0 and is included in the box $[-1, 1]^n$.

We further consider a model where the total production of each period is a "here-and-now" decision. Conversely, the specific dispatching of the products to the periods (represented by y), the stock, backlog and spoiling are "wait-and-see" decisions; that is, they can be adjusted to past realizations of the demand vector ξ and become decision functions $s : \Xi \to \mathbb{R}^n_+$, $r : \Xi \to \mathbb{R}^n_+$, and $\pi : \Xi \to \mathbb{R}^n_+$. To prevent the decision maker to take decisions based on the realization of future events, the functions s and r must satisfy the so-called non-anticipativity constraints, which are stated next. Given any n-dimensional vector v, we denote its projection over the first i components by $v(i) := (v_1, \dots, v_i)$. The non-anticipativity constraints can be formally defined as follows.

$$s_i(\xi) = s_i(\xi') \quad \forall \xi, \xi' \in \Xi; \xi(i) = \xi'(i), \tag{10}$$
$$r_i(\xi) = r_i(\xi') \quad \forall \xi, \xi' \in \Xi; \xi(i) = \xi'(i), \tag{11}$$
$$\pi_i(\xi) = \pi_i(\xi') \quad \forall \xi, \xi' \in \Xi; \xi(i) = \xi'(i), \tag{12}$$
$$y_{ji}(\xi) = y_{ji}(\xi') \quad \forall \xi, \xi' \in \Xi; \xi(i) = \xi'(i). \tag{13}$$

These constraints impose that if the scenarios ξ and ξ' coincide for the first i time periods, then the decisions taken until that period must be the same for both scenarios. In particular, (13) means that the dispatching of the demand of period i among all time periods is decided at period i.

A common practice in inventory management of perishable products is to follow a FIFO policy, that is, the demand is satisfied with the oldest products. This policy can be ensured by the additional set of bilinear constraints.

$$y_{ki}(\xi)\pi_j(\xi) = 0 \quad \forall j \in H; k > j - m; i \leq j; \xi \in \Xi. \tag{14}$$

Constraints (14) ensure that $y_{ki}(\xi)$ and $\pi_j(\xi)$ cannot be simultaneously positive. A positive value for $\pi_j(\xi)$ means that a product was produced in time period $j-m$ and was not used during its shelf life. A positive value for $y_{ki}(\xi)$ represents a product produced after time period $j-m$ (the time period a spoiled item in j was produced) that is used to fulfill demand until period j. Such demand could be satisfied with a spoiled item whose shelf life ended in j if $\pi_j(\xi) > 0$.

The robust counterpart of (1)–(8), including non-anticipativity constraints and the FIFO restrictions follows.

$$
(F1)\quad
\begin{aligned}
&\min z && && (15)\\
&\text{s.t. } z \geq \sum_{i\in H}(c_i x_i + h_i s_i(\xi) + p_i r_i(\xi) + q_i \pi_i(\xi)), && \forall \xi \in \varXi && (16)\\
&s_i(\xi) = \sum_{j=1}^{i} \sum_{k=i+1}^{\min(i+m,n)} y_{jk}(\xi) + \sum_{j=i+1}^{\min(i+m,n)} \pi_j(\xi), && \forall i \in H, \xi \in \varXi && (17)\\
&r_i(\xi) = \sum_{j=i+1}^{n+1} \sum_{k=1}^{i} y_{jk}(\xi), && \forall i \in H, \xi \in \varXi && (18)\\
&x_i = \sum_{j=1}^{i+m} y_{ij}(\xi) + \pi_{i+m}(\xi), && \forall i \in \{1,\dots,n-m\}, \xi \in \varXi && (19)\\
&x_i = \sum_{j=1}^{n+1} y_{ij}(\xi), && \forall i \in \{n-m+1,\dots,n\}, \xi \in \varXi && (20)\\
&\sum_{i=\max(1,j-m)}^{n+1} y_{ij}(\xi) = d_j(\xi), && \forall j \in H, \xi \in \varXi && (21)\\
&(10)-(14) && &&\\
&x \in X && && (22)\\
&x, y, \pi, r, s \geq 0 && && (23)
\end{aligned}
$$

where z is a new variable representing the worst-case cost, following the usual epigraph reformulation. Contrasting with the deterministic situation, we provide below an example showing that the robust problem can have optimal solutions involving non-zero spoiling functions π.

Example 1 *Consider the nominal demand vector $\overline{d} = (1,1,0,0)$, and deviations $\overline{d} = (1,1,1,1)$ assume the correlation matrix $\overline{D}$ is the identity matrix, $m = 2$ and the uncertainty set is given by the budget polytope $\varXi = \{\xi \in \mathbb{R}^n : \sum_{i\in H} |\xi_i| \leq \varGamma, |\xi_i| \leq 1, i \in H\}$ with $\varGamma = 1$. Assume $c_i = 2, h_i = 1, p_i = 10, q_i = 2$. Then, in order to prevent backlog (since its cost is very high), the optimal policy is $x = (2,1,0,0)$. The worst scenario is $\xi^* = (-1,0,0,0)$ which corresponds to the case where demand is as low as possible. Hence $d(\xi^*) = (0,1,0,0)$. For this scenario the optimal solution is given by $y_{12}(\xi^*) = 1, \pi_2(\xi^*) = 1$ and $\pi_3(\xi^*) = 1$.*

Observe that, considering $p_2 = 1$ and $p_i = 2$, for $i \neq 2$ the optimal solution without imposing the FIFO policy would be $y_{22}(\xi^*) = 1, \pi_2(\xi^*) = 2$.

The above problem contains infinitely many constraints and variables, including the infinite number of non-convex FIFO constraints (14). To our knowledge, no algorithmic approach from the literature is able to address such a problem exactly, even for small instances. Fortunately, we show in the following Section that is possible to reformulate $(F1)$ as another non-linear infinite problem, having simpler non-linear constraints. More importantly, we show in Section 4 that the reformulation is compatible with row-and-column generation algorithms.

3 Reformulation

Let us first show how variables y can be removed from formulation $(F1)$. To simplify notations, the following formulations also use variables $\pi_1, \dots, \pi_m$ whose values are always equal to 0.

$$
\begin{aligned}
& \min z && \\
(F2) \quad & \text{s.t. } z \geq \sum_{i \in H} (c_i x_i + h_i s_i(\xi) + p_i r_i(\xi) + q_i \pi_i(\xi)) && \forall \xi \in \Xi \quad (24)\\
& s_i(\xi) \geq \sum_{k=1}^{i} (x_k - d_k(\xi) - \pi_k(\xi)) && \forall i \in H, \xi \in \Xi \quad (25)\\
& r_i(\xi) \geq \sum_{k=1}^{i} (d_k(\xi) + \pi_k(\xi) - x_k) && \forall i \in H, \xi \in \Xi \quad (26)\\
& \pi_{i+m}(\xi) = \max\left(0, \sum_{k=1}^{i} x_k - \sum_{k=1}^{i+m} d_k(\xi) - \sum_{k=m+1}^{i+m-1} \pi_k(\xi)\right) && \forall i \in \{1, \dots, n-m\}, \xi \in \Xi \quad (27)\\
& \pi_i(\xi) = 0 && \forall i \in \{1, \dots, m\}, \xi \in \Xi \quad (28)\\
& x, s, r \geq 0 && (29)
\end{aligned}
$$

We will prove the following result.

Theorem 1 *A vector (x, y, z, r, s, π) is an optimal solution to $(F1)$ if and only if (x, z, r, s, π) is an optimal solution to $(F2)$.*

To prove the result, we introduce first a useful property satisfied by the feasible solutions of $(F1)$.

Lemma 2 *The following flow balance constraints hold for each feasible solution of $(F1)$*

$$
x_j + s_{j-1}(\xi) + r_j(\xi) = d_j(\xi) + \pi_j(\xi) + s_j(\xi) + r_{j-1}(\xi), \quad \forall j \in H, \tag{30}
$$

where $s_0(\xi) = r_0(\xi) = 0$.

Proof The result follows from substituting x_i, $s_{i-1}(\xi)$ and $r_i(\xi)$ with the rhs of constraints (17), (18), and (19), respectively, and re-arranging the resulting summations.

Below we prove that the optimal solutions to $(F1)$ are feasible for $(F2)$.

Lemma 3 *If (x, y, s, r, π) is optimal for $(F1)$, it satisfies (25) – (27).*

Proof Summing up equations (30) from $j = 1$ to $j = i$ it follows that

$$
r_i(\xi) + \sum_{j=1}^{i} x_j = \sum_{j=1}^{i} d_j(\xi) + \sum_{j=1}^{i} \pi_j(\xi) + s_i(\xi). \tag{31}
$$

Combining (31) with the non-negativity of r_i and s_i, we obtain

$$
s_i(\xi) \geq \sum_{j=1}^{i} (x_j - d_j(\xi) - \pi_j(\xi)) \quad \forall j \in H \tag{32}
$$

$$
r_i(\xi) \geq \sum_{j=1}^{i} (d_j(\xi) - x_j + \pi_j(\xi)) \quad \forall j \in H \tag{33}
$$

proving that (25) and (26) are satisfied.

Algorithm 1: Constructing y

```
for $\xi \in \Xi$ do
    $i, j \leftarrow 1$;
    $\overline{d}_i \leftarrow d_i(\xi)$;
    $\overline{x}_j \leftarrow x_j$;
    while $i < n + 1$ do
        if $i \leq j + m$ then
            $y_{ji}(\xi) \leftarrow \max(\overline{x}_j, \overline{d}_i)$;
            $\overline{d}_i(\xi) \leftarrow \max(0, \overline{d}_i - \overline{x}_j)$;
            $\overline{x}_j \leftarrow \max(0, \overline{x}_j - \overline{d}_i)$;
        else
            $\pi_{j+m} \leftarrow \overline{x}_j$;
            $\overline{x}_j \leftarrow 0$;
        end
        if $\overline{d}_i = 0$ then $i \leftarrow i + 1$; $\overline{d}_i \leftarrow d_i(\xi)$;
        if $\overline{x}_j = 0$ then $j \leftarrow j + 1$; $\overline{x}_j \leftarrow x_j$;
    end
end
return: $y$
```

Concerning (27), let $i \leq n - m$. Summing up equations (19) from $j = 1$ to $j = i$, we obtain

$$\begin{aligned}\sum_{j=1}^{i} x_j &= \sum_{j=1}^{i} \sum_{k=1}^{j+m} y_{jk}(\xi) + \sum_{j=1}^{i} \pi_{j+m}(\xi) && (34)\\ &= \sum_{k=1}^{i+m} \sum_{j=\max(1,k-m)}^{i} y_{jk}(\xi) + \sum_{j=m+1}^{i+m} \pi_j(\xi) && (35)\\ &= \sum_{k=1}^{i+m} \left(d_k(\xi) - \sum_{j=i+1}^{n+1} y_{jk}(\xi) \right) + \sum_{j=m+1}^{i+m} \pi_j(\xi). && (36)\end{aligned}$$

Isolating $\pi_{i+m}(\xi)$ in the lhs, we obtain

$$\pi_{i+m}(\xi) = \sum_{k=1}^{i} x_k - \sum_{k=1}^{i+m} d_k(\xi) + \sum_{k=1}^{i+m} \sum_{j=i+1}^{n+1} y_{jk}(\xi) - \sum_{k=m+1}^{i+m-1} \pi_k(\xi). \tag{37}$$

Notice at this point that the summation of variables y in (37) represents the production from periods $\{i+1, \ldots, n+1\}$ to periods $\{1, \ldots, i+m\}$. Two cases occur depending on the value of that summation. If the summation is zero, then (37) becomes

$$\pi_{i+m}(\xi) = \sum_{k=1}^{i} x_k - \sum_{k=1}^{i+m} d_k(\xi) - \sum_{k=m+1}^{i+m-1} \pi_k(\xi). \tag{38}$$

and the rhs is non-negative because of the non-negativity restriction on π_{i+m} in $(F1)$. Otherwise, the summation is positive meaning that the production from periods $\{i+1, \ldots, n+1\}$ to periods $\{1, \ldots, i+m\}$ is positive. In that case, constraints (14) imply

$$\pi_{i+m}(\xi) = 0. \tag{39}$$

Grouping the cases (38) and (39), we obtain (27).

Let us now turn to optimal solutions to $(F2)$ and show that they can be extended to feasible solutions of $(F1)$.

Lemma 4 *If (x, s, r, π) is optimal for $(F2)$, we can define a vector y so that (x, y, s, r, π) be feasible for $(F1)$.*

Proof Since we consider an optimal solution, we may assume that s and r satisfy the restrictions (25) and (26) tightly. For each $\xi \in \Xi$, let us define the vector $y(\xi)$ by iteratively using the production described x_j to satisfy the demand $d_i(\xi)$, thus setting the value of $y_{ji}(\xi)$. Whenever we reach i and j such that $i > j + m$, the production in excess is affected to π_{j+m}. The construction is formally described in Algorithm 1. One readily verifies that the values for s, r, and π given by equalities (25)–(27) are equal to those resulting from Algorithm 1. Moreover, the values provided for y satisfy the non-anticipativity restrictions because the value set for $y_{ji}(\xi)$ depends only on the demands $\{d_1(\xi), \dots, d_i(\xi)\}$.

4 Row-and-column generation algorithm

Problem $(F2)$ contains infinite numbers of variables and constraints, making it intractable as such. Here, we tackle the problem by alternating between solving a relaxed master problem and separation problems. While being still non-linear, the master problem is a *finite* optimization problem, contrasting with the infinite optimization problems discussed in the previous sections. Let $\mathcal{S} \subseteq \Xi$ be the finite set generated so far through the separation problems and let us define the relaxed master problem as

$$
\begin{aligned}
& \min z \\
(F2\text{-}RM) \quad & \text{s.t. } z \geq \sum_{i \in H} (c_i x_i + h_i s_i(\xi) + p_i r_i(\xi) + q_i \pi_i(\xi)) && \forall \xi \in \mathcal{S} && (40)\\
& s_i(\xi) \geq \sum_{k=1}^{i} (x_k - d_k(\xi) - \pi_k(\xi)) && \forall i \in H, \xi \in \mathcal{S} && (41)\\
& r_i(\xi) \geq \sum_{k=1}^{i} (d_k(\xi) + \pi_k(\xi) - x_k) && \forall i \in H, \xi \in \mathcal{S} && (42)\\
& \pi_{i+m}(\xi) = \max\left(0, \sum_{k=1}^{i} x_k - \sum_{k=1}^{i+m} d_k(\xi) - \sum_{k=m+1}^{i+m-1} \pi_k(\xi)\right) && \forall i \in \{1, \dots, n-m\}, \xi \in \mathcal{S} && (43)\\
& \pi_i(\xi) = 0 && \forall i \in \{1, \dots, m\}, \xi \in \mathcal{S} && (44)\\
& x, s, r \geq 0 && && (45)
\end{aligned}
$$

Given a solution (x^*, z^*) for $(F2\text{-}RM)$ the separation problem determines if there exists a vector $\xi \in \Xi \backslash \mathcal{S}$ for which the associated constraints (40)–(43) are violated. As already mentioned, (41) and (42) are satisfied at equality in any optimal solution. Therefore, the separation problem amounts to verify whether there exists $\xi \in \Xi$ such that the

$$\sum_{i \in H} (c_i x_i + h_i s_i(\xi) + p_i r_i(\xi) + q_i \pi_i(\xi)) > z^*$$

where s, r and π are given by (41)–(45). This can be reformulated as the following non-linear maximization problem in variables (ξ, s, r, π), where variables (s, r, π) are redundant variables used only to simplify the problem description

$$\begin{aligned}
&& \max \sum_{i \in H} (h_i s_i + p_i r_i + q_i \pi_i) && && (46)\\
&(F2\text{-}sep) & \text{s.t. } \xi \in \Xi && && (47)\\
&& s_i = \max\left(0, \sum_{k=1}^{i} (x_k^* - d_k(\xi) - \pi_k)\right) && \forall i \in H && (48)\\
&& r_i = \max\left(0, \sum_{k=1}^{i} (d_k(\xi) + \pi_k - x_k^*)\right) && \forall i \in H && (49)\\
&& \pi_{i+m} = \max\left(0, \sum_{k=1}^{i} x_k^* - \sum_{k=1}^{i+m} d_k(\xi) - \sum_{k=m+1}^{i+m-1} \pi_k\right) && \forall i \in \{1, \dots, n-m\} && (50)\\
&& \pi_i = 0 && \forall i \in \{1, \dots, m\}. && (51)
\end{aligned}$$

Let ξ^* and ω^* denote the optimal solution to $(F2\text{-}sep)$ and its cost, respectively. If $\omega^* > z^* - \sum_{i \in H} c_i x_i^*$, then the optimal vector ξ^* is added to $\mathcal{S}$, leading to the addition of the corresponding constraints (40)–(45) and variables $s(\xi^*), r(\xi^*), \pi(\xi^*)$. Otherwise, the current solution (x^*, z^*) to $(F2\text{-}RM)$ is optimal.

Finally, the non-linearities in $(F2\text{-}sep)$ can be handled using classical techniques that introduce binary variables and large coefficients denoted by M. First, we introduce the real and binary variables u_i and α_i, respectively, and replace each restriction of (50) with

$$\begin{aligned}
& u_{i+m} = \sum_{k=1}^{i} x_k^* - \sum_{k=1}^{i+m} d_k(\xi) - \sum_{k=m+1}^{i+m-1} \pi_k && (52)\\
& \pi_{i+m} \leq M\alpha_{i+m}, && (53)\\
& \pi_{i+m} \geq u_{i+m} && (54)\\
& \pi_{i+m} \leq u_{i+m} + M(1 - \alpha_i) && (55)\\
& \alpha_{i+m} \in \{0, 1\} && (56)
\end{aligned}$$

Second, we introduce the real variables v_i, w_i, z_i and binary variables β_i, replace each pair of restrictions (48) and (49) with

$$\begin{aligned}
& v_i = \sum_{k=1}^{i} (x_k^* - d_k(\xi) - \pi_k) && (57)\\
& w_i = \sum_{k=1}^{i} (d_k(\xi) + \pi_k - x_k^*) && (58)\\
& z_i \leq h_i v_i + M\beta_i && (59)\\
& z_i \leq p_i w_i + M(1 - \beta_i) && (60)\\
& \beta_i \in \{0, 1\}, && (61)
\end{aligned}$$

and replace $h_i s_i + p_i r_i$ by z_i in the objective function (46).

5 Computational results

This section presents some of the computational experiments carried out to test the performance of the row-and-column generation algorithm and to provide a sensitivity analysis of the perishable production as function of some parameters.

All tests were run on a computer with processor Intel(R) Core(TM) i7, CPU 3.20GHz, with 8GB of RAM using the optimization software Cplex studio 12.7.

5.1 Data generation

The data is generated in order to test a wide range of practical cases. Regarding the cost structure, three sets of instances are considered: `Dynamic`, `Static` and `Random`.

The costs from instance set `Dynamic` have a given periodicity, representing the seasonality of certain products, and are computed as follows:

$$
\begin{aligned}
c_i &= 10 + 5\text{sin}(\tfrac{15.i.\pi}{180}), i \in H,\\
h_i &= 2 + 1\text{sin}(\tfrac{15.i.\pi}{180}), i \in H,\\
p_i &= 50 + 25\text{sin}(\tfrac{15.i.\pi}{180}), i \in H,\\
\bar{d}_i &= 1000 + 500\text{sin}(\tfrac{15.i.\pi}{180}), i \in H,\\
q_i &= \beta + \beta\sin(\tfrac{15.i.\pi}{180}), i \in H.
\end{aligned}
$$

Instances from the set `Static` represent instances with fixed costs and are generated as follows:

$$
\begin{aligned}
c_i &= 20, i \in H\\
h_i &= 4, i \in H,\\
p_i &= 100, i \in H,\\
\bar{d}_i &= 1000, i \in H,\\
q_i &= \beta, i \in H.
\end{aligned}
$$

Finally, the costs of instances from set `Random` represent instance where the costs vary randomly. These costs are randomly generated, using a uniform distribution, as follows:

$$
\begin{aligned}
c_i &= 10 + 10\tfrac{\xi+1}{10}, i \in H,\\
h_i &= 2 + 2\tfrac{\xi+1}{10}, i \in H,\\
p_i &= 30 + 30\tfrac{\xi+1}{10}, i \in H,\\
\bar{d}_i &= 1000 + 1000\tfrac{\xi+1}{10}, i \in H,\\
q_i &= 1 + \beta\tfrac{\xi+1}{10}, i \in H.
\end{aligned}
$$

where ξ represents a random integer between 0 and 9. As we aim to study the spoiled production in detail, the spoiled cost is also controlled by a parameter β that can have three possible values: 2, 20 and 200. The number of time periods considered, n, belongs to the set $\{10, 20, 30, 40, 50\}$. For the production capacity two cases are considered, the constant capacity case with $C_i = 5000, \forall i \in H$ and the unbounded case $C_i = \infty, \forall i \in H$. For the maximum allowed deviations in the client demands we consider $\hat{d}_i = \alpha\bar{d}_i$ where parameter α can take values in $\{0.1, 0.2, 0.3\}$.

Finally, we emphasize that such choice of parameters allows us to simulate close-to-reality instances while providing flexibility in the cost structures. In real cases, the cost of storage is, in general, lower than the cost to produce and to dispose a product. The backlog costs are usually high since they penalize customer dissatisfaction associated with fulfil demand with delay.

5.2 Optimization approach analysis

# Periods	Capacity	Total Time	Master Time	Adv. Time	# Iterations
10	yes	156.262	134.824	21.437	10.358
10	no	146.509	129.368	17.140	9.502
20	yes	283.326	226.850	56.475	8.198
20	no	151.118	102.567	48.550	8.000
30	yes	543.358	362.400	180.957	10.016
30	no	487.562	340.824	146.737	9.601
40	yes	784.884	508.278	276.605	11.387
40	no	789.871	531.912	257.958	10.852
50	yes	1050.207	668.398	381.808	12.037
50	no	1017.892	642.535	375.368	11.642

Table 1 Average running times and average numbers of iterations of the decomposition approach.

As expected the running times increases when the number of periods increases. For the largest size instances the average running times are around 1000 seconds (less than half an hour). Considering the capacities, we can observe that the running times tend to be a bit higher for the smaller instances. Notice the impact of restricting the production capacity would be expected to be more relevant in the master problem. Considering the running times spent in each of the two subproblems, we see that largest amount of time is spent with the master problem. However, the increase of the running times with the increase of time periods is faster with the adversarial problem than with the master problem. This can be easily understood since the adversarial problem considers several big-M constraints, which are known to produce bad duality bounds. The average number of iterations is relatively low, ranging from 8 to 12.

Next we detail the running times according to the shelf-life and cost structure. From Figure 1 we can observe that the running times are higher for short shelf-life items, which may indicate that the robust inventory problem with perishable products may be computationally more difficult to solve than the corresponding inventory problem where product deterioration with time is not considered. Regarding the cost structure, there is no clear trend for $n = 10, 20, 30$. However, for $n = 40$ and $n = 50$ the running times are clearly higher for the set of instances with the `dynamic` cost structure.

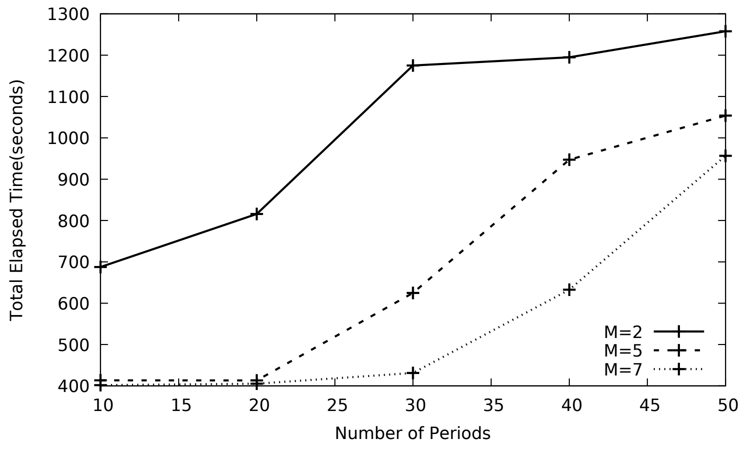


Fig. 1 Running times as a function of products shelf-life.

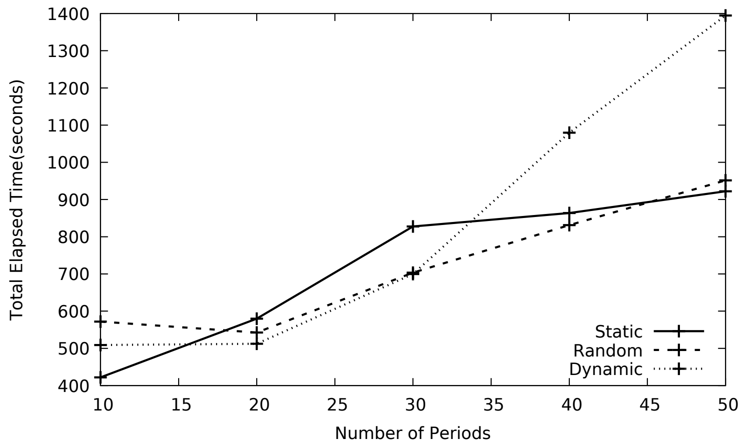


Fig. 2 Running times as a function of the number of periods.

5.3 Sensitivity analysis for the spoiled production

Here we report the tests conducted to evaluate the impact of the uncertainty parameters Γ and β on the spoiled amount when the worst-case scenario occurs. Each possible Γ and β pair represents a degree of uncertainty. For large values of Γ and β we assume a large degree of uncertainty (both on the total number of demand that deviate from their nominal value as the amount of each individual deviation). Conversely, for small values of these parameters the uncertainty becomes very restricted. Γ varies in $\{1, 3, 5\}$ and β in $\{0.01, 0.05, 0.1\}$ giving a total number of 9 combinations, labeled from A, ($\Gamma = 1, \beta = 0.01$), lowest conservative case, to I, ($\Gamma = 5, \beta = 0.1$), highest conservative case. In Figure 3 we report the percentage of amount of production that is lost due to the end of the shelf-life of product in case the worst-case scenario is materialized for the optimal solution found. The results are split according to the instance cost structure (dynamic, static and random) and shelf-life, $m = 2, 5, 7$.

As expected, when m increases the amount of production lost decreases. With a shelf-life of $m = 7$ periods, the lost production is very low, only in the most adversarial case we can observe a loss higher than 1%. Moreover, for this case, the production losses occur mainly for the `dynamic` cost structure (top charts layer). On the other hand, the short shelf-life of 2 periods, the robust approach is able to find solutions that for the worst the case scenario and under the highest degree of uncertainty, ensure a production loss around 9% of the total demand. Additionally, if we restrict the variation of one of the parameter assignments $\Gamma = 5$ or $\beta = 0.1$, the production loss drops for less than 4%.

5.4 Sensitivity analysis for the cost structures

In this section, we take a closer look at the impact that variations in the costs has in the optimal value, elapsed time and spoiled amount of our proposed method. Since the results for the three cost structures considered here are quite similar, and since the `dynamic` cost structure generates the hardest instances and, simultaneously, is the cost structure that that better simulates real instances, we restrain ourselves to the `dynamic` cost structure. We consider 4 variants of the `dynamic` cost structure, named `D1`, `D2`, `D3`, `D4`, and characterized as follows:

`D1` Instances (Standard Instance)

$$\begin{aligned} c_i &= 10 + 5\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ h_i &= 2 + 1\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ p_i &= 50 + 25\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ \bar{d}_i &= 1000 + 500\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ q_i &= \beta + \beta\sin(\tfrac{15.i.\pi}{180}), i \in H. \end{aligned}$$

`D2` Instances (Low Backlog)

$$\begin{aligned} c_i &= 10 + 5\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ h_i &= 2 + 1\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ p_i &= 20 + 10\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ \bar{d}_i &= 1000 + 500\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ q_i &= \beta + \beta\sin(\tfrac{15.i.\pi}{180}), i \in H. \end{aligned}$$

`D3` Instances (High Production Cost)

$$\begin{aligned} c_i &= 20 + 10\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ h_i &= 2 + 1\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ p_i &= 50 + 25\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ \bar{d}_i &= 1000 + 500\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ q_i &= \beta + \beta\sin(\tfrac{15.i.\pi}{180}), i \in H. \end{aligned}$$

`D4` Instances (High Storage Cost)

$$\begin{aligned} c_i &= 10 + 5\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ h_i &= 10 + 5\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ p_i &= 50 + 25\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ \bar{d}_i &= 1000 + 500\sin(\tfrac{15.i.\pi}{180}), i \in H,\\ q_i &= \beta + \beta\sin(\tfrac{15.i.\pi}{180}), i \in H. \end{aligned}$$

As said before, these new instances represent different variations of the `dynamic` cost structure. The instances in the set `D1` are the standard instances of the set `dynamic`, the instances in the set `D2` consider a lower backlog cost, the instances in the set `D3` consider a higher production cost than those in the standard `dynamic` instances and the set `D4` considers a higher storage cost than those in the standard `dynamic` instances.

This new set of computational experiments were carried out on a computer with processor Intel(R) Core(TM) i7, CPU 2.70GHz with 8GB of RAM using the optimization software Cplex studio 12.7.

5.4.1 Elapsed Time and Optimal Value

First we take a look on the way the variations at the costs affects the total elapsed time and the optimal value of the proposed method.

As expected, the optimal value grows proportionally with the number of periods for all the sets of instances, almost linearly. The instances that present a higher production cost and storage cost, **D3** and **D4** respectively, present a higher optimal value as we can see in Figure 4.

The same behavior cannot be observed for the total running time. Figure 5 shows that, although the total running time is increasing, the increase in the storage cost produces a more unstable problem. We claim that the increase in the storage cost can make the spoiled amount more relevant, hence those instances whose optimal solutions may have large spoiled amounts become more difficult.

5.4.2 Spoiled Amount

We take a look at the effects of the variation of the costs in the spoiled amount. Figure 6 presents the spoiled amount grouped by the value of the uncertainty parameter Γ and Figure 7 presents the spoiled amount grouped by the value of the shelf life of the product. We can notice that, overall, the largest spoiled amount occurs for instances `D1` and `D4`. We also notice that the increase of the uncertainty parameter Γ produces an increase in the spoiled amount, while the increase of the shelf life parameter M produces a decrease in the spoiled amount. This happens because high uncertainty may lead to worst case scenarios where large amounts of products exceed their shelf life and, conversely, a large shelf life provides a broader horizon to use the product, hence minimizing the spoiled amount.

6 Conclusions

We have considered a robust inventory problem where products are perishable and are handled through a FIFO policy. We have introduced a robust model and a non-linear reformulation, which is solved through a row-and-column generation algorithm.

Computational tests have been conducted to cover a broad class of instances simulating different realistic cases. These tests have shown that (i) the solution approach can solve to optimality, and within reasonable amount of running time, all the tested instances; (ii) the robust solutions obtained are able to ensure, for the worst-case scenarios, low production losses due to the end of the product shelf life; (iii) the instances considering short shelf life were computationally harder to solve, which seems to indicate that the inclusion of products perishability adds some degree of complexity to the inventory problems.

This study raises two questions for future research. From a theoretical point of view, it would be interesting to understand whether the inclusion of perishability changes the complexity of the problem or, at least, if it changes the complexity of the separation problem. From a practical point of view, it would be interesting to compare the solutions obtained using the proposed robust approach with the solutions resulting from stochastic models assuming different probability distributions for the demands.

Acknowledgments

We would like to thank the referees for suggestions leading to improving the presentation of the manuscript.

References

1. A. Agra, M. C. Santos, D. Nace, and M. Poss. A dynamic programming approach for a class of robust optimization problems. *SIAM Journal on Optimization*, 26(3):1799–1823, 2016.
2. P. Amorim, H. Meyr, C. Almeder, and B. Almada-Lobo. Managing perishability in production-distribution planning: a discussion and review. *Flexible Services and Manufacturing Journal*, 25(3):389–413, 2013.
3. J. Ayoub and M. Poss. Decomposition for adjustable robust linear optimization subject to uncertainty polytope. *Computational Management Science*, 13(2):219–239, 2016.
4. M. Bakker, J. Riezebos, and R. H. Teunter. Review of inventory systems with deterioration since 2001. *European Journal of Operational Research*, 221(2):275 – 284, 2012.
5. A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski. Adjustable robust solutions of uncertain linear programs. *Math. Program.*, 99(2):351–376, 2004.
6. D. Bertsimas, D. B. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM review*, 53(3):464–501, 2011.
7. D. Bertsimas and I. Dunning. Multistage robust mixed-integer optimization with adaptive partitions. *Operations Research*, 64(4):980–998, 2016.
8. D. Bertsimas and A. Thiele. A robust optimization approach to inventory theory. *Operations Research*, 54(1):150–168, 2006.
9. D. Bienstock and N. Özbay. Computing robust basestock levels. *Discrete Optimization*, 5(2):389 – 414, 2008. In Memory of George B. Dantzig.
10. A. Billionnet, M. Costa, and P. Poirion. 2-stage robust MILP with continuous recourse variables. *Discrete Applied Mathematics*, 170:21–32, 2014.
11. D. Bogataj, M. Bogataj, and D. Hudoklin. Mitigating risks of perishable products in the cyber-physical systems based on the extended mrp model. *International Journal of Production Economics*, 193:51 – 62, 2017.
12. L. C. Coelho and G. Laporte. Optimal joint replenishment, delivery and inventory management policies for perishable products. *Computers & Operations Research*, 47:42 – 52, 2014.
13. E. Delage and D. Iancu. Robust multistage decision making. *Tutorials in Operations Research*, 2015.

14. B. L. Gorissen and D. den Hertog. Robust counterparts of inequalities containing sums of maxima of linear functions. *European Journal of Operational Research*, 227(1):30–43, 2013.
15. B. L. Gorissen, I. Yanıkoglu, and D. den Hertog. A practical guide to robust optimization. *Omega*, 53:124–137, 2015.
16. A. Gutierrez-Alcoba, G. Ortega, E. Hendrix, and I. García. Accelerating an algorithm for perishable inventory control on heterogeneous platforms. *Journal of Parallel and Distributed Computing*, 104:12 – 18, 2017.
17. A. Gutierrez-Alcoba, R. Rossi, B. Martin-Barragan, and E. Hendrix. A simple heuristic for perishable item inventory control under non-stationary stochastic demand. *International Journal of Production Research*, 55(7):1885–1897, 2017.
18. A. Herbon. Should retailers hold a perishable product having different ages? the case of a homogeneous market and multiplicative demand model. *International Journal of Production Economics*, 193:479 – 490, 2017.
19. V. N. Hsu. An economic lot size model for perishable products with age-dependent inventory and backorder costs. *IIE Transactions*, 35(8):775–780, 2003.
20. L. Janssen, T. Claus, and J. Sauer. Literature review of deteriorating inventory models by key topics from 2012 to 2015. *International Journal of Production Economics*, 182:86 – 112, 2016.
21. M. A. Kaasgari, D. M. Imani, and M. Mahmoodjanloo. Optimizing a vendor managed inventory (vmi) supply chain for perishable products by considering discount: Two calibrated meta-heuristic algorithms. *Computers & Industrial Engineering*, 103:227 – 241, 2017.
22. D. Kuhn, W. Wiesemann, and A. Georghiou. Primal and dual linear decision rules in stochastic and robust optimization. *Math. Program.*, 130(1):177–209, 2011.
23. C. C. Lee. Two-warehouse inventory model with deterioration under fifo dispatching policy. *European Journal of Operational Research*, 174(2):861–873, 2006.
24. S. Nahmias. Perishable inventory theory: A review. *Operations Research*, 30(4):680–708, 1982.
25. S. Nahmias. *Perishable Inventory Systems.* International Series in Operations Research & Management Science. Springer, 2011.
26. J. Pahl and S. Voss. Integrating deterioration and lifetime constraints in production and supply chain planning: A survey. *European Journal of Operational Research*, 238(3):654 – 674, 2014.
27. K. Postek and D. den Hertog. Multistage adjustable robust mixed-integer optimization via iterative splitting of the uncertainty set. *INFORMS Journal on Computing*, 28(3):553–574, 2016.
28. M. C. Santos, M. Poss, and D. Nace. A perfect information lower bound for robust lot-sizing problems. *Annals of Operations Research*, 271(2), 887-913, 2017.
29. I. Yanıkoglu, B. L. Gorissen, and D. den Hertog. A survey of adjustable robust optimization. *EJOR.* In press.
30. B. Zeng and L. Zhao. Solving two-stage robust optimization problems by a constraint-and-column generation method. *Operations Research Letters*, 41(5):457–461, 2013.

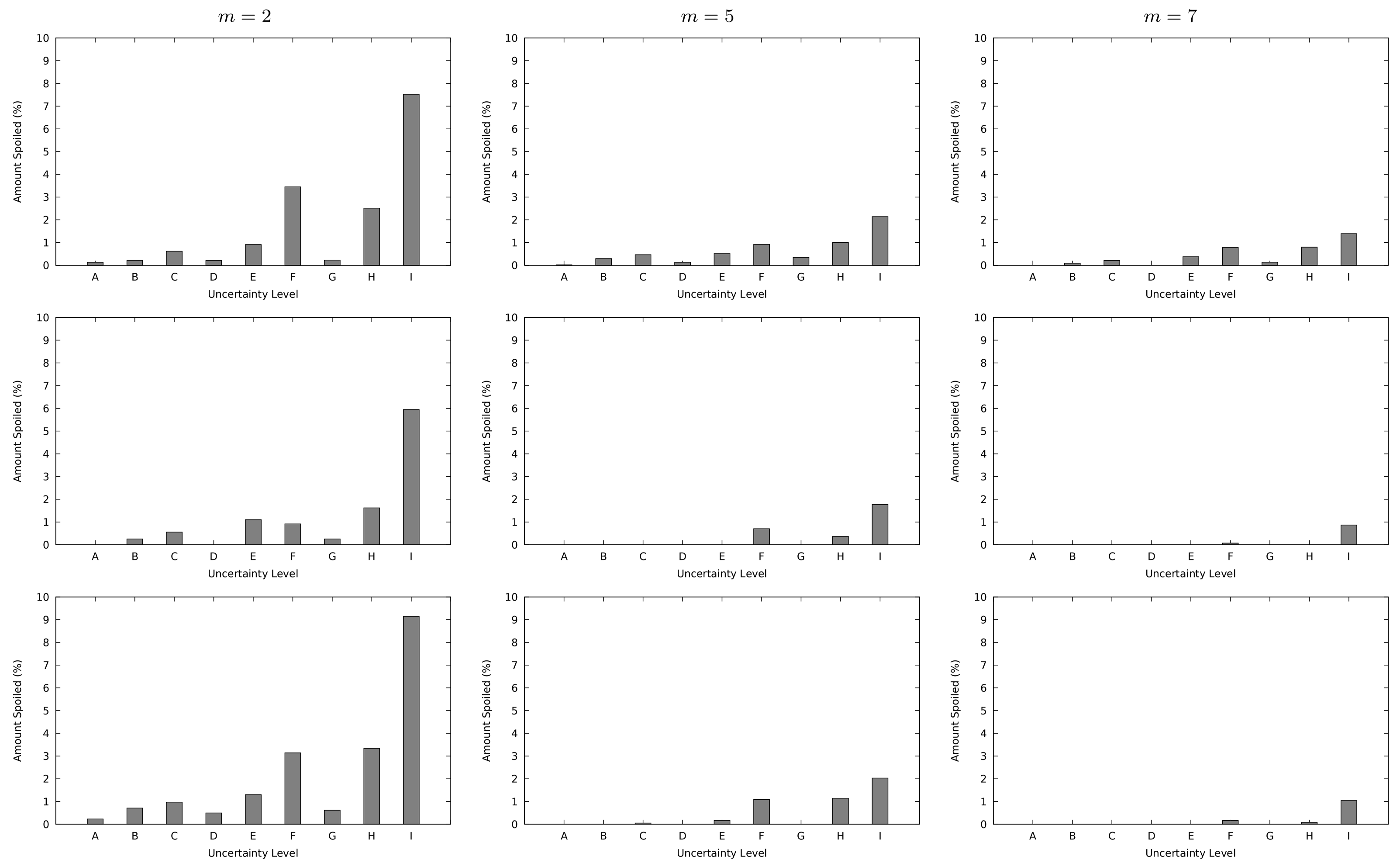
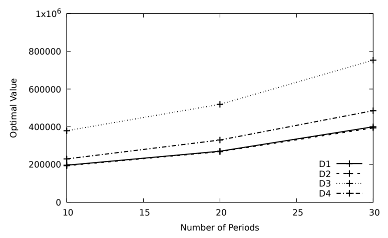
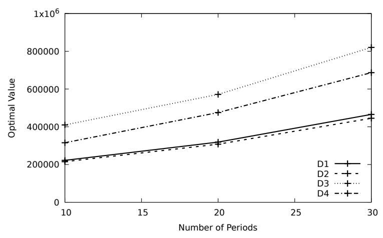


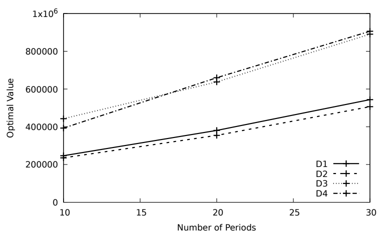
Fig. 3 Spoiled products *versus* degree of uncertainty for instances with dynamic (top charts layer), static (medium charts layer), random (bottom charts layer) and costs. For each layer three graphics are presented for $m = 2, 5, 7$ (from left to right). Each graphic has nine bars corresponding to the following parameters controlling uncertainty: A: $\Gamma = 1$ and $\beta = 0.01$; B: $\Gamma = 1$ and $\beta = 0.05$; C: $\Gamma = 1$ and $\beta = 0.1$; D: $\Gamma = 3$ and $\beta = 0.01$; E: $\Gamma = 3$ and $\beta = 0.05$; F: $\Gamma = 3$ and $\beta = 0.1$; G: $\Gamma = 5$ and $\beta = 0.01$; H: $\Gamma = 5$ and $\beta = 0.05$; I: $\Gamma = 5$ and $\beta = 0.1$.



(a) $\mathit{\Gamma} = 1$

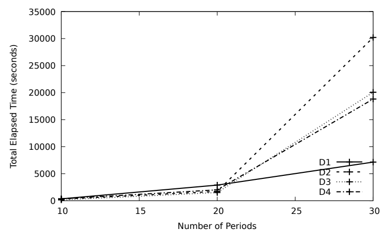


(b) $\mathit{\Gamma} = 3$

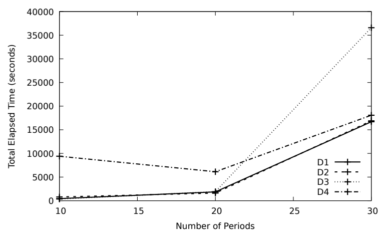


(c) $\mathit{\Gamma} = 5$

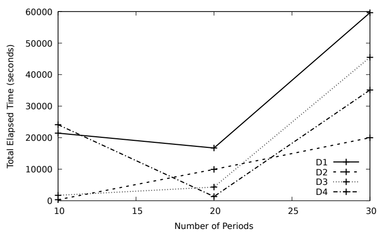
Fig. 4 Optimal value grouped by the value of the parameter $\mathit{\Gamma}$.



(a) $\mathit{\Gamma} = 1$

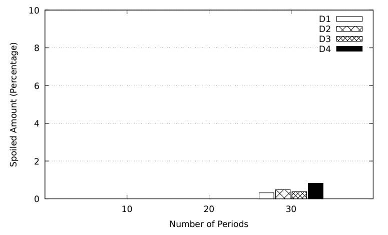


(b) $\mathit{\Gamma} = 3$

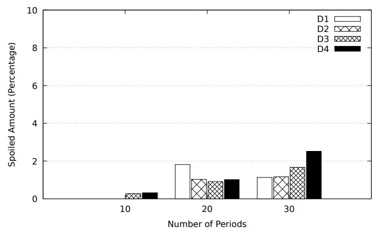


(c) $\mathit{\Gamma} = 5$

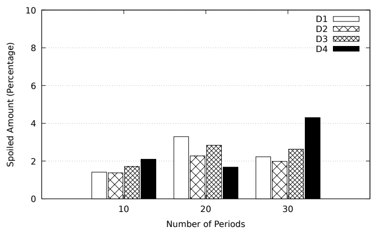
Fig. 5 Running time elapsed on the problem grouped by the value of the parameter $\mathit{\Gamma}$.



(a) $\mathit{\Gamma} = 1$

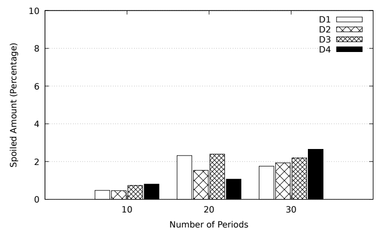


(b) $\mathit{\Gamma} = 3$

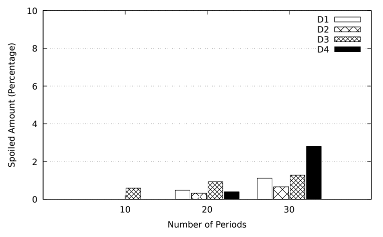


(c) $\mathit{\Gamma} = 5$

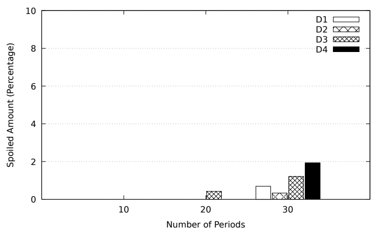
Fig. 6 Spoiled amount grouped by the value of the parameter $\mathit{\Gamma}$.



(a) $M = 2$



(b) $M = 5$



(c) $M = 7$

Fig. 7 Spoiled amount grouped by the value of the parameter M.