Computing non-stationary (s, S) policies using mixed integer linear programming

Mengyuan Xiang[a], Roberto Rossi[a,*], Belen Martin-Barragan[a], S. Armagan Tarim[b]

[a] *Business School, University of Edinburgh, Edinburgh, United Kingdom*
[b] *Department of Management, Cankaya University, Ankara, Turkey*

Abstract

This paper addresses the single-item single-stocking location stochastic lot sizing problem under the (s, S) policy. We first present a mixed integer non-linear programming (MINLP) formulation for determining near-optimal (s, S) policy parameters. To tackle larger instances, we then combine the previously introduced MINLP model and a binary search approach. These models can be reformulated as mixed integer linear programming (MILP) models which can be easily implemented and solved by using off-the-shelf optimisation software. Computational experiments demonstrate that optimality gaps of these models are around 0.3% of the optimal policy cost and computational times are reasonable.

Keywords: supply chain management, (s, S) policy, stochastic lot-sizing, mixed integer programming, binary search

1. Introduction

Stochastic lot sizing is an important research area in inventory theory. One of the landmark studies is Scarf (1960) which proved the optimality of (s, S) policies for a class of dynamic inventory models. The (s, S) policy features two

*Corresponding author
Email addresses: mengyuan.xiang@ed.ac.uk (Mengyuan Xiang), roberto.rossi@ed.ac.uk (Roberto Rossi), belen.martin-barragan@ed.ac.uk (Belen Martin-Barragan), at@cankaya.edu.tr (S. Armagan Tarim)

control parameters: s and S. Under this policy, the decision maker checks the opening inventory level at the beginning of each time period: if it drops to or below the reorder point s, then a replenishment should be placed to reach the order-up-to-level S. Unfortunately, computing optimal (s, S) policy parameters remains a computationally intensive task.

In the literature, studies on (s, S) policy can be categorized into stationary and non-stationary. A number of attempts have been made to compute stationary (s, S) policy parameters, e.g. (Iglehart, 1963; Veinott Jr and Wagner, 1965; Archibald and Silver, 1978; Stidham Jr, 1977; Sahin, 1982; Federgruen and Zipkin, 1984; Zheng and Federgruen, 1991; Feng and Xiao, 2000). However, in reality, there has been an increasing recognition that lot-sizing studies need to be undertaken for non-stationary environments (Graves, 1999). Additionally, only two studies investigated computations of (s, S) policy under non-stationary stochastic demand (Askin, 1981; Bollapragada and Morton, 1999). This motivates our work on non-stationary (s, S) policy.

Askin (1981) adopted the "least cost per unit time" approach in selecting order-up-to-levels and reorder points under a penalty cost scheme. Decision makers first determine desired cycle lengths and order-up-to-levels. Then, reorder points are decided by means of a trade-off analysis between expected costs per period in cases of ordering and not ordering.

As Bollapragada and Morton (1999) pointed out, Askin (1981) is computationally expensive because of the convolutions of demand distributions. In contrast, Bollapragada and Morton (1999) proposed a stationary approximation heuristic for computing optimal (s, S) policy parameters. Firstly, decision makers precompute pairs of (s, S) values for various demand parameters and tabulate results. Here, a large number of efficient algorithms exist for generating the stationary table, e.g. (Federgruen and Zipkin, 1984; Zheng and Federgruen, 1991; Feng and Xiao, 2000). Secondly, order-up-to-levels and reorder points can be read from stationary tables by averaging the demand parameters over an estimate of the expected time between two orders. However, this algorithm relies upon complex code, particularly for generating stationary tables.

Unfortunately, both these works (Askin, 1981; Bollapragada and Morton, 1999) do not provide a satisfactory solution to the problem: they rely on ad-hoc computer coding and provide relatively large optimality gaps. A recent computational study Dural-Selcuk et al. (2016) estimated the optimality gap of (Askin, 1981; Bollapragada and Morton, 1999) at 3.9% and 4.9%, respectively. These drawbacks motivate our work in finding a heuristic method for computing (s, S) policy parameters which does not need computer coding and can provide better optimality gaps.

In this paper, we therefore introduce a new modelling framework to compute near-optimal (s, S) policy parameters. In particular, we consider a single-item single-stocking location stochastic lot-sizing problem under non-stationary demand, fixed and unit ordering cost, holding cost and penalty cost. In contrast to other approaches in the literature, our models can be easily implemented and solved by using off-the-shelf software such as IBM ILOG optimisation studio. We make the following contributions to literature on stochastic lot-sizing.

- We introduce the first mixed integer non-linear programming (MINLP) model to compute near-optimal (s, S) policy parameters.
- We show that this model can be reformulated as a mixed integer linear programming (MILP) model by piecewise linearising the cost function; this reformulation can be solved by using off-the-shelf software.
- To tackle larger instances, we combine the previously introduced MINLP model and a binary search procedure.
- Computational experiments demonstrate that optimality gaps of our models are tighter than existing algorithms (Askin, 1981; Bollapragada and Morton, 1999) in the literature, and computational times of our models are reasonable.

The rest of this paper is organised as follows. Section 2 describes problem settings and a stochastic dynamic programming (SDP) formulation. Section 3 discusses the notion of K-convexity and introduces relevant K-convex cost

functions which are approximated by an MINLP model in Section 4. Section 5 presents an MINLP heuristic for approximating (s, S) policy parameters. Section 6 introduces an alternative binary search approach for computing (s, S) policy parameters. A detailed computational study is given in Section 7. Finally, we draw conclusions in Section 8.

2. Problem description

We consider a single-item single-stocking location inventory management system over a T-period planning horizon. We assume that orders are placed at the beginning of each time period, and delivered instantaneously. There exist ordering costs $c(\cdot)$ comprising a fixed ordering cost K for placing an order, and a linear ordering cost c proportional to order quantity Q. Demands d_t in each period $t = 1, \ldots, T$ are independent random variables with known probability distributions. At the end of period t, a linear holding cost h is charged on every unit carried from one period to the next; and a linear penalty cost b is occurred for each unmet demand at the end of each time period.

For a given period $t = \{1, \ldots, T\}$, let I_{t-1} denote the opening inventory level and Q_t represent the order quantity. Then the immediate cost of period t can be expressed as

$$f_t(I_{t-1}, Q_t) = c(Q_t) + \mathrm{E}[h \max(I_{t-1} + Q_t - d_t, 0) + b \max(d_t - I_{t-1} - Q_t, 0)], \quad (1)$$

where E denotes the expectation taken with respect to the random demand d_t. Additionally, the ordering cost $c(Q_t)$ is defined as:

$$c(Q_t) = \begin{cases} K + c\ Q_t, & Q_t > 0 \\ 0, & Q_t = 0 \end{cases}$$

Let $C_t(I_{t-1})$ represent the expected total cost of an optimal policy over periods $t, \ldots, T$ when the initial inventory level at the beginning of period t is I_{t-1}. We model the problem as a stochastic dynamic program (Bellman, 1957) via the following functional equation

$$C_t(I_{t-1}) = \min_{Q_t} \{ f_t(I_{t-1}, Q_t) + \mathrm{E}[C_{t+1}(I_{t-1} + Q_t - d_t)] \} \quad (2)$$

where

$$C_T(I_{T-1}) = \min_{Q_t} f_T(I_{T-1}, Q_T)$$

represents the boundary condition.

3. The optimality of (s, S) policies in stochastic lot sizing

Scarf (1960) proved that the optimal policy in the dynamic inventory problem is always of the (s, S) type based on a study of the function

$$G_t(y) = cy + \mathrm{E}[h \max(y - d_t, 0) + b \max(d_t - y, 0)] + \mathrm{E}[C_{t+1}(y - d_t)], \quad (3)$$

where y is the stock level immediately after purchases are delivered.

Since we consider a non-stationary environment, values of the (s, S) policy parameters will depend on the given period t. Let (s_t, S_t) denote the policy parameters for period t. Function $G_t(y)$ can be used to define the policy parameters (s_t, S_t) and prove their optimality. In particular, the order-up-to-level S_t is defined as the value minimising $G_t(y)$; whereas the parameters s_t is given by the value $s_t < S_t$ such that $K + G_t(S_t) = G_t(s_t)$. K-convexity of the function $G_t(y)$ ensures the uniqueness of s_t and S_t (Scarf, 1960).

Example. We illustrate the concepts introduced on a 4-period example. Demand d_t is normally distributed in each period t with mean $\mu_t \in \{20, 40, 60, 40\}$, for $t = 1, \ldots, 4$ respectively. Standard deviation σ_t of demand in period t is equal to $0.25\mu_t$. Other parameters are $K = 100$, $h = 1$, $b = 10$, and $c = 0$. We plot $G_1(y)$ in Fig. 1 for initial inventory levels $y \in (0, 200)$. The expected total costs $G_1(y)$ are obtained via SDP. The order-up-to-level is $S_1 = 70$ and the minimised expected total cost $G_1(S_1) = 262.5839$; the reorder point is $s_1 = 14$ and the corresponding cost $G_1(s_1) = 362.5839$. Note that $G_1(s_1) = G_1(S_1) + K$. The optimal policy is to order to 70 if the initial inventory $y < 14$; otherwise not to order.

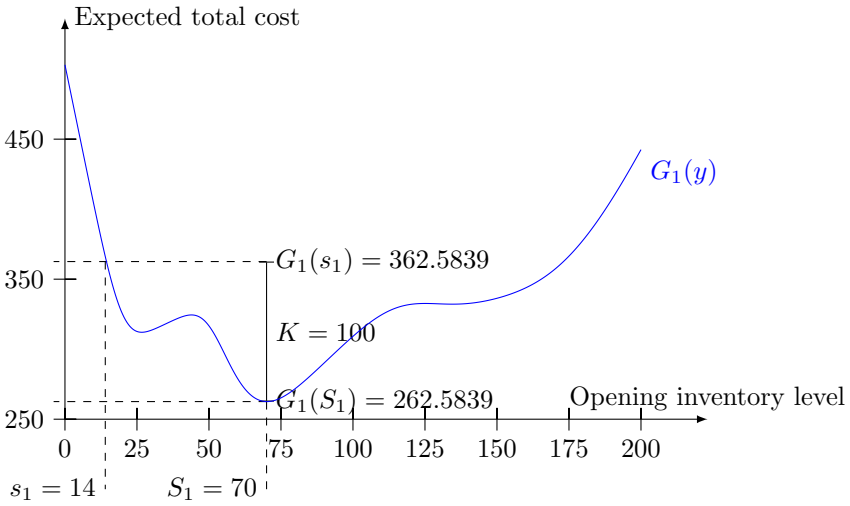

Figure 1: Plot of $G_1(y)$

4. MINLP approximation of Scarf's $G_t(y)$ function

In this section, we exploit an MINLP model to approximate the function $G_t(y)$ in Eq. (3). Our model follows the control policy known as "static-dynamic uncertainty" strategy, originally introduced in Bookbinder and Tan (1988). Under this strategy, the timing of orders and order-up-to-levels are expected to be determined at the beginning of the planning horizon, while associated order quantities are decided upon only when orders are issued. As illustrated in Rossi et al. (2015), this strategy provides a cost performance which is close to the optimal "dynamic uncertainty" strategy. However, optimal (s, S) parameters cannot be immediately derived from existing mathematical programming models operating under a static-dynamic uncertainty strategy, such as Tarim and Kingsman (2006), and Rossi et al. (2015). We next illustrate how a model operating under a static-dynamic uncertainty strategy can be used to approximate the function $G_t(y)$ in Eq. (3).

Consider a random variable ω and a scalar variable x. The first order loss function is defined as $L(x, \omega) = \mathrm{E}[\max(\omega - x, 0)]$, where E denotes the expected value with respect to the random variable ω. The complementary first order

loss function is defined as $\hat{L}(x, \omega) = \mathrm{E}[\max(x - \omega, 0)]$. Like Rossi et al. (2015), we will model non-linear holding and penalty costs by means of this function.

Consider three sets of decision variables: $\tilde{I}_t$, the expected closing inventory level at the end of period t, with I_0 denoting the initial inventory level; δ_t, a binary variable which is set to one if an order is placed in period t; P_{jt}, a binary variable which is set to one if and only if the most recent replenishment before period t was issued in period j. Let $\tilde{d}_{jt}$ denote the expected value of the demand over periods $j, \ldots, t$, i.e. $\tilde{d}_{jt} = \tilde{d}_j + \cdots + \tilde{d}_t$. Decision variables $H_t \geq 0$ and $B_t \geq 0$ for $t = 1, \ldots, T$ represent end of period t expected excess inventory and back-orders, respectively. An MINLP formulation for the non-stationary stochastic lot-sizing problem, obtained following the modeling strategy in Rossi et al. (2015), is shown in Figure 2.

$$\min \big(-cI_0 + c\sum_{t=1}^{T} \tilde{d}_t + \sum_{t=1}^{T} (K\delta_t + hH_t + bB_t) + c\tilde{I}_T \big) \tag{4}$$

Subject to, $t = 1, 2, \ldots, T$

$$\delta_t = 0 \rightarrow \tilde{I}_t + \tilde{d}_t - \tilde{I}_{t-1} = 0 \tag{5}$$

$$\sum_{j=1}^{t} P_{jt} = 1 \tag{6}$$

$$P_{jt} \geq \delta_j - \sum_{k=j+1}^{t} \delta_k, \qquad j = 1, 2, \ldots, t \tag{7}$$

$$P_{jt} = 1 \rightarrow H_t = \hat{L}(\tilde{I}_t + \tilde{d}_{jt}, d_{jt}), \qquad j = 1, 2, \ldots, t \tag{8}$$

$$P_{jt} = 1 \rightarrow B_t = L(\tilde{I}_t + \tilde{d}_{jt}, d_{jt}), \qquad j = 1, 2, \ldots, t \tag{9}$$

$$P_{jt} \in \{0, 1\}, \qquad j = 1, 2, \ldots, t \tag{10}$$

$$\delta_t \in \{0, 1\} \tag{11}$$

Figure 2: The formulation of the non-stationary stochastic lot-sizing problem

The objective function (4) computes the minimised expected total cost com-

prising ordering cost, holding cost and penalty cost. Constraints (5) state inventory balance equations. Constraints (6) indicate the most recent replenishment before period t was issued in period j. Constraints (7) identify uniquely the period in which the most recent replenishment prior to t took place. Constraints (8) and (9) model end of period t expected excess inventory and back-orders by means of the first order loss function.

We now discuss how to adapt the model in Fig. 2 in order to approximate $G_t(y)$. We call this modified model MINLP-s, and use superscript "s" to label decision variables in this model. For any given initial inventory level I_0^s, let $G_1^s(I_0^s)$ denote the expected total cost over periods $1,\ldots,T$ without issuing an order in period 1,

$$G_1^s(I_0^s) = -cI_0^s + c\sum_{t=1}^{T}\tilde{d}_t + \sum_{t=1}^{T}(K\delta_t^s + hH_t^s + bB_t^s) + c\tilde{I}_T^s. \tag{12}$$

MINLP-s optimises $G_1^s(I_0^s)$ subject to constraints in Fig. 2 with an additional constraint

$$\delta_1^s = 0, \tag{13}$$

which forces the model not to place a replenishment in period 1. Note that MINLP-s can easily be approximated as an MILP model by using the approach discussed in Rossi et al. (2015) to piecewise linearise loss functions in constraints (8) and (9). For further details please refer to Appendix A.

Example. In Fig. 3, we plot the expected total cost $G_1^s(y)$ for the same 4-period numerical example in Fig. 1 with initial inventory level $I_0^s \in (0, 200)$, $G_1^s(y)$ are obtained via the MILP-s. Since $G_1^s(y)$ approximates $G_1(y)$, we can use $G_1^s(y)$ to find approximate values $\hat{S}_1$ and $\hat{s}_1$ for S_1 and s_1.

单独从 G(y)也能搜到 s，S。但计算太麻烦。下面构造了一个联合 MIP 模型

5. An MINLP-based model to approximate (s, S) policy parameters

In this section we present an MINLP heuristic for computing near-optimal (s, S) policy parameters. To the best of our knowledge, this is the first MINLP model for computing near-optimal (s, S) policy parameters.

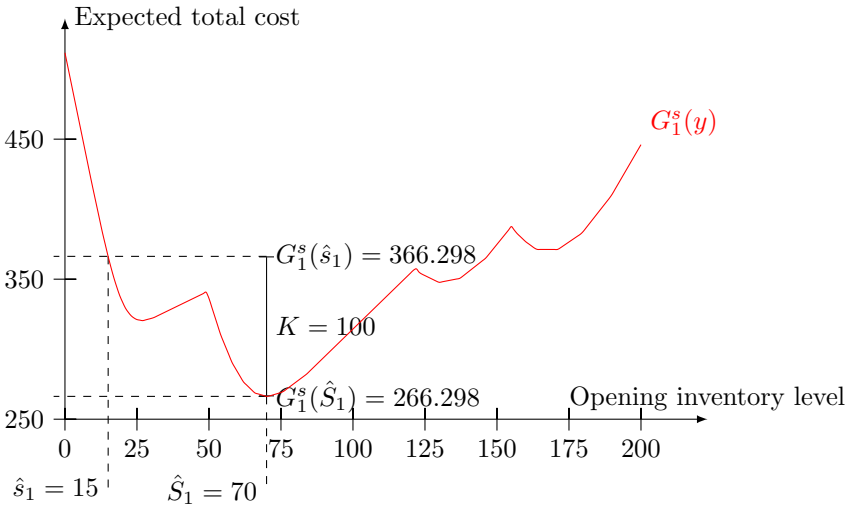

Figure 3: Plot of $G_1^s(y)$

In a similar fashion to "MINLP-s", we introduce "MINLP-S". MINLP-S imposes the constraint

强制订货,这个很巧

$$\delta_1^S = 1, \tag{14}$$

which forces the model to place a replenishment in period 1. Similarly to Eq. 12, let the objective function of MINLP-S be $C_1^S(\cdot)$, which approximates $C_1(\cdot)$.

不是通过理论分析，而是通过观察

Recall that I_0^S represents the initial inventory level in MINLP-S. Since in MINLP-S a replenishment is forced in period 1 (Eq. 14), this variable — which is left free to vary in the model — represents an approximation $\hat{S}_1$ of the order-up-to-level S_1 in period 1. We observe that $C_1^S(\hat{S}_1) = G_1^s(\hat{S}_1) + K$, since the only difference between MINLP-S and MINLP-s is the constraint that prescribes whether to force or not a replenishment in period 1.

强制订货，总补货到 S

通过观察，其实只要 x <= S，都有 C(x)= G(S) + K - v(S-x)，因此 G(s) = C(S)

Since $G_1^s(y)$ is an approximation of $G_1(y)$, if we identify an opening inventory level $I_0^s < \hat{S}_1$ such that $G_1^s(I_0^s) = G_1^s(I_0^S) + K$, then $s_1 \approx I_0^s$. Therefore, we can approximate s_1 and S_1 simultaneously by connecting MINLP-S and MINLP-s via the constraint

重要

通过这个条件将两个模型联合起来

$$G_1^s(I_0^s) = C_1^S(I_0^S). \tag{15}$$

Finally, since $s_1 \leq S_1$, we introduce an additional constraint to ensure that the reorder point is not greater than the order-up-to-level,

$$I_0^s \leq I_0^S. \tag{16}$$

Note that, in contrast to the true value $G_1(y)$, there is no guarantee that K-convexity holds for its approximation $G_1^s(y)$. For some instances we may therefore have multiple values s_1 such that (15) holds. As we will demonstrate in our computational study, leaving to the solver the freedom to choose one of such values in a non-deterministic fashion leads to competitive results.

MINLP-S and MINLP-s are connected by Eq. (15), in such a way the order-up-to-level S_1, the reorder point s_1, and the optimal expected total cost are approximated simultaneously. For the joint MINLP model, decision variables are those in both MINLP-S and MINLP-s with addition of initial inventory levels I_0^S and I_0^s. The holistic objective function is to minimise the expected total cost of MINLP-S over the planning horizon and the expected total cost of MINLP-s from period two to the end of the planing horizon,

$$\begin{aligned}\min \Big(& -cI_0^S + c\sum_{t=1}^{T} \tilde{d}_t^S + \sum_{t=1}^{T}(K\delta_t^S + hH_t^S + bB_t^S) + c\tilde{I}_T^S \\ & -cI_0^s + c\sum_{t=1}^{T} \tilde{d}_t^s + \sum_{t=2}^{T}(K\delta_t^s + hH_t^s + bB_t^s) + c\tilde{I}_T^s \Big);\end{aligned} \tag{17}$$

note that the missing period for MINLP-s is taken care of by constraints 13 and 15.

Constraints of the joint MINLP model are those of both MINLP-S and MINLP-s in addition to the linking constraints (13), (14), (15) and (16). By solving the joint MINLP model over the planning horizon $k, \ldots, T$, one estimates S_k and s_k, where $k = 1, \ldots, T$. As previously discussed, the joint MINLP model can also be linearised via the piecewise-linear approximation proposed in Rossi et al. (2015). In our MILP model, (8) and (9) are modelled via the piecewise OPL expression (IBM, 2011). For a complete overview of the MILP model refer to Appendix B.

Example. We now use the same 4-period numerical example in Fig. 3 to demonstrate the modelling strategy behind the joint MINLP heuristic. We observe that, for period 1, the approximated order-up-to-level is $S_1 = 70.2658$, the reorder point is $s_1 = 15.0008$, the optimal expected total cost $G_1^s(s_1) = 366.138$ as shown in Fig. 1. By solving the joint MNILP repeatedly, S_t, s_t and $G_t^s(s_t)$, for $t = 1, \ldots, 4$, are estimated as shown in Table 1.

t	1	2	3	4
s_t	15.0008	29.0161	58.1089	29.0161
S_t	70.2658	53.9768	116.5530	53.9768
$G_t^s(s_t)$	366.138	311.369	193.338	118.031

Table 1: Near-optimal (s, S) policy parameters obtained via the joint MINLP heuristic

6. A binary search approach to approximate (s, S) policy parameters

The joint MINLP heuristic presented in the last section can only effectively tackle small-size instances. In order to tackle larger-size problems, we introduce a more efficient approach that combines the model MINLP-s discussed in Section 5 and a binary search strategy. More precisely, we first let I_0^s to be a decision variable in MINLP-s and minimise $G_k^s(I_0^s)$ to estimate the order-up-to-level $\hat{S}_1$ and the minimised expected total cost $G_1^s(\hat{S}_1)$ for period 1. Next, since the K-convexity holds for $G_1(y)$, there exits a unique reorder point s_1 such that $G_1(s_1) = G_1(S_1) + K$. Since $G_1^s(I_0^s)$ is an approximation of $G_1(y)$, we can conduct a binary search to approximate the reorder point $\hat{s}_1$ by $I_0^s \leq \hat{S}_1$ at which $G_1^s(I_0^s) = G_1^s(\hat{S}_1) + K$. By repeating this procedure over the planning horizon $k, \ldots, T$, we find pairs of S_k and s_k, where $k = 1, \ldots, T$.

Algorithm 1 shows the binary search approach. For any given planning horizon $k, \ldots, T$, where $k = 1, \ldots, T$, we first let I_{k-1}^s to be a decision variable in MINLP-s and minimise $G_k^s(I_{k-1}^s)$ so that to estimate the order-up-to-level $\hat{S}_k$ and the minimised expected total cost $G_k^s(\hat{S}_k)$ for period k. We assume, for the binary search method, the initial low value (low) is a large negative integer

and the initial high value ($high$) is equal to $\hat{S}_k$ (line 4 in Algorithm 1). Then, we start the binary search procedure (line 5) while $low < high$. We calculate the average value $mid = low + \text{round}((high - low)/2)$ (Line 6). Next step is to run the MINLP-s by updating the initial inventory level I^s_{k-1} with the calculated middle value $I^s_{k-1} = mid$ and to obtain current expected total cost $G^s_k(I^s_{k-1})$ (line 8). If current cost $G^s_k(I^s_{k-1}) - G^s_k(\hat{S}_k) - K < 0$, then we update $high = low - stepsize$ (line 10); if current cost $G^s_k(I^s_{k-1}) - G^s_k(\hat{S}_k) - K > 0$, then we update $low = mid + stepsize$ (line 12); otherwise, $\hat{s}_k = mid$ (line 14). By repeating this procedure over planning horizon $k, \ldots, T$, we obtain $\hat{s}_k$, $\hat{S}_k$, and the optimal cost, where $k = 1, \ldots, T$.

一定能保证搜到的是最小的s?
可能会搜到一个较大的 s

第一个初始下界要足够小

Example. We illustrate the solution method just discussed via the same 4-period numerical example presented in Fig. 1. We assume the step size of the binary search is 0.01. We observe that the order-up-to-level $\hat{S}_1 = 70.2658$ and the expected total cost $G^s_1(70.2658) = 266.298$. We then set $low = -200$, $high = 70.2658$. While $low < high$, the mid is updated via the comparison of $G^s_1(I^s_0)$ and $G^s_1(70.2658)$. After a number of iterations, we obtain the reorder point $\hat{s}_1 = 15$ at which $G^s_1(15) = G^s_1(70.2658)$. By repeating this procedure we obtain $\hat{S}_t$, $\hat{s}_t$, and $G^s_t(s_t)$, for each period $t = 1, \ldots, 4$ as displayed in Table 2.

7. Computational experience

In this section we present an extensive analysis of the heuristics discussed in Sections 5 (MP) and 6 (BS). We first design a test bed featuring instances defined over an 8-period planning horizon. On this test bed, we assess the behaviour of the optimality gap and the computational efficiency of both the MP and BS heuristics. Then we assess the computational performance of our the BS heuristics on a test bed featuring larger instances on a 25-period planning horizon. For all cases, MINLP models are solved by employing the piecewise linearization strategy discussed in Rossi et al. (2015), which can be easily implemented in OPL by means of the `piecewise` syntax. Numerical examples are conducted by using the IBM ILOG CPLEX Optimization Studio 12.7 and MATLAB R2014a

8 个阶段

Data: costs (ordering cost, holding cost, penalty cost), mean demand and standard deviation of each period, stepsize

Result: pairs of s and S for each period

for $k = 1$ **to** T **do**
  Minimising MINLP-s in Section 5 in OPL;
  Obtaining $G_k^s(\hat{S}_k)$ and $\hat{S}_k$;
  low = a large negative integer; $high = \hat{S}_k$;
  while $low < high$ **do**
    $mid = low + \text{round}((high - low)/2)$;
    Running the MINLP-s with $I_{k-1}^s =$ in OPL;
    Obtaining currentcost $G_k^s(I_{k-1}^s)$;
    if $G_k^s(I_{k-1}^s) - G_k^s(\hat{S}_k) - K < 0.0001$ **then**
      $high = mid -$ stepsize;
      else if $G_k^s(I_{k-1}^s) - G_k^s(\hat{S}_k) - K > 0.0001$ **then**
        $low = mid +$ stepsize;
        else
          $\hat{S}_k = mid$;
          $low = high$;
        end
      end
    end
  end
end

Algorithm 1: The binary search algorithm

on a 3.2GHz Intel(R) Core(TM) with 8GB of RAM.

7.1. An 8-period test bed

We consider a test bed which includes 270 instances. Specifically, we incorporate ten demand patterns displayed in Fig. 4. These patterns comprising two

t	1	2	3	4
s_t	15	29.01	58.1	29.01
S_t	70.2658	53.9768	116.5530	53.9768
$G_t^s(s_t)$	366.138	311.369	193.338	118.031

Table 2: Near-optimal (s, S) policy parameters obtained via the binary search approach

life cycle patterns (LCY1 and LCY2), two sinusoidal patterns (SIN1 and SIN2), a stationary pattern (STA), a random pattern (RAND), and four empirical patterns (EMP1, ..., EMP4). Full details on the experimental setup are given in Appendix C. Fixed ordering cost K ranges in $\{200, 300, 400\}$, the penalty cost b takes values $\{5, 10, 20\}$. We assume that demand d_t in each period t is independent and normally distributed with mean $\tilde{d}_t$ and coefficient of variation $c_v \in \{0.1, 0.2, 0.3\}$; note that $\sigma_t = c_v \tilde{d}_t$. Since we operate under the assumption of normality, our models can be readily linearised by using the piecewise linearisation parameters available in Rossi et al. (2014). However, the reader should note that our proposed modeling strategy is distribution independent, see Rossi et al. (2015).

We set the SDP model discussed in Section 2 as a benchmark. We compare against this benchmark in terms of optimality gap and computational time. First of all, we obtain optimal parameters for each test instance by implementing an SDP algorithm in MATLAB. Then, we solve each instance by adopting both modelling heuristics presented in Section 5 and 6. Specifically, for the MP heuristic we employ six segments in the piecewise-linear approximations of B_t and H_t (for $t = 1, \ldots, T$) in order to guarantee reasonable computational performances; for the BS heuristic, whose computational performance is only marginally affected by an increased number of segments in the linearisation, we employ eleven segments and a step size 0.1. To estimate the cost of the policies obtained via our heuristics, we simulate all policies via Monte Carlo Simulation (10,000 replications).

Table 7.1 gives an overview of optimality gaps in terms of modelling methods

Figure 4: Demand patterns in our computational analysis

and parameter settings. Both heuristics perform better when demand pattern is rather steady. It is difficult to make a general remark with respect to fixed ordering cost. Both methods perform worse as penalty cost increases. More specifically, when penalty cost increases from 10 to 20, the optimal gap rises from 0.28% to 0.38% and from 0.25% to 0.44%, respectively. Similarly, performance of these two methods deteriorates as demand variability increases: optimality gap of the BS heuristic increases significantly from 0.18% to 0.39% as the coefficient of variation increases from 0.1 to 0.3. Overall, the average optimality gap of the MP heuristic is 0.33%, and that of the BS method is 0.28%. This discrepancy ought to be expected, since in the case of the BS method a higher number of segments has been employed.

Existing heuristics Askin (1981) and Bollapragada and Morton (1999) were reimplemented by Dural-Selcuk et al. (2016) and assessed on a test bed that neatly resembles the one adopted in this work. As shown in Dural-Selcuk et al. (2016), Askin's optimality gap is 3.9%, and Bollapragada and Morton's is 4.9%. The optimality gap of our heuristic is 0.33% when six segments are employed in the piecewise linearisation, and it drops to 0.28% when eleven segments are employed. Our models therefore outperform both Askin (1981) and Bollapragada and Morton (1999) in terms of optimality gap on the test bed here considered.

Table 4 shows computational times with regard to different setting parameters and modelling methods. Note "STDEV" in Table 4 represents the standard deviation. The average computational time of the MP heuristic is $51.01s$, that of the BS method is $7.64s$, and that of the SDP model is $60.21s$. The computational times of the SDP and of the MP model vary significantly for different demand patterns considered, while that of the BS method remains stable. In particular, when the demand setting is EMP3, the average computational time of the MP model is $286.21s$; whereas, when the demand setting is EMP4, it is just $22.41s$. We observe that fixed ordering cost, penalty cost, and coefficient of variation do not have considerable influence on computational time of small-scale instances. Additionally, standard deviation of the MP model and of the SDP model fluctuate significantly, while that of the BS tend to remain stable.

Modelling methods	MP	BS
Demand pattern		
LCY1	0.28	0.39
LCY2	0.26	0.15
SIN1	0.18	0.14
SIN2	0.17	0.16
STA	0.25	0.23
RAND	0.14	0.16
EMP1	0.41	0.36
EMP2	1.01	0.78
EMP3	0.17	0.17
EMP4	0.44	0.21
Fixed ordering cost		
200	0.32	0.28
300	0.29	0.20
400	0.38	0.34
Penalty cost		
5	0.19	0.14
10	0.28	0.25
20	0.38	0.44
Coefficient of variation		
0.1	0.22	0.18
0.2	0.32	0.25
0.3	0.46	0.39
Average gap	0.33	0.28

Table 3: Average optimality gaps % of the 8-period test bed with different setting parameters and modelling methods

Settings	MP		BS		SDP	
	Mean	STDEV	Mean	STDEV	Mean	STDEV
Demand pattern						
LCY1	4.07	0.81	8.22	0.66	14.42	0.03
LCY2	25.73	47.36	8.15	0.76	14.41	0.03
SIN1	3.88	0.74	6.90	0.64	14.41	0.02
SIN2	3.85	0.62	6.70	0.70	14.37	0.08
STA	9.18	21.13	6.84	0.63	7.69	0.05
RAND	3.48	0.51	7.48	1.07	7.50	0.06
EMP1	53.32	140.72	8.00	0.82	150.13	1.12
EMP2	97.99	162.94	8.17	0.77	114.44	1.31
EMP3	286.21	636.73	7.49	0.72	114.46	1.09
EMP4	22.41	40.25	8.45	0.89	150.24	0.35
Fixed ordering cost						
200	88.81	365.45	7.71	0.97	60.17	59.96
300	33.75	99.76	7.62	0.92	60.29	60.07
400	30.48	109.84	7.59	1.07	60.16	59.99
Penalty cost						
5	81.62	343.68	7.44	0.93	60.34	60.14
10	51.78	182.68	7.56	0.86	60.24	60.03
20	19.63	65.62	7.92	1.06	60.04	59.83
Coefficient of variation						
0.1	39.22	165.33	7.66	1.00	60.23	60.01
0.2	76.09	348.42	7.66	0.91	60.18	59.98
0.3	37.73	89.68	7.59	1.05	60.20	60.03
Average	51.01	51.01	7.64	0.99	60.21	60.21

Table 4: Average computational times (seconds) of the 8-period test bed with different setting parameters and modelling methods

7.2. A 25-period test bed

As shown in Section 7.1 for the 8-period test bed, both the MP and the BS methods provide tight optimality gaps and acceptable computational efficiency. We now extend the 8-period test bed to 25 periods with larger instances. Demands of LCY1, LCY2, SIN1, SIN2, STA, and RAND are generated with expressions (18), (19), (20), (21), (22), and (23) in Fig. 5. Demands of EMP1, EMP2, EMP3 and EMP4 are derived from Strijbosch et al. (2011). Full details are given in Appendix C. Assume that fixed ordering cost ranges in $\{500, 1000, 1500\}$, penalty cost takes values $\{5, 10, 20\}$, and the coefficients of standard deviations are $\{0.1, 0.2, 0.3\}$.

$$d_t = \text{round}\left(\frac{190 \times e^{-(t-13)^2}}{2 \times 5^2}\right), \qquad t = 1, 2, \ldots, T \qquad (18)$$

$$d_t = \text{round}\left(\frac{170 \times e^{-(t-13)^2}}{2 \times 6^2}\right), \qquad t = 1, 2, \ldots, T \qquad (19)$$

$$d_t = \text{round}\Big(70 \times \sin(0.8t) + 80\Big), \qquad t = 1, 2, \ldots, T \qquad (20)$$

$$d_t = \text{round}\Big(30 \times \sin(0.8t) + 100\Big), \qquad t = 1, 2, \ldots, T \qquad (21)$$

$$d_t = 100, \qquad t = 1, 2, \ldots, T \qquad (22)$$

$$d_t = \text{round}(\text{random}(0, 250)), \qquad t = 1, 2, \ldots, T \qquad (23)$$

Figure 5: Expressions for generating demand data

We obtain optimal (s, S) parameters and record computational times obtained via the BS method. For the first 15 periods we perform binary search with step size 1 in order to ensure fast convergence; for the last 10 periods, we adopt a step size 0.1 to enhance accuracy. The number of segments used in the piecewise linearisation is eleven. To estimate the cost of the policy obtained via our approximation, we simulate each instance one million times in MATLAB. We summarise computational times in Table 5.

According to Table 5, the computational time drops dramatically from 1039.40s

Settings	Mean	standard deviation
Demand pattern		
LCY1	588.18	213.91
LCY2	806.25	338.10
SIN1	579.45	181.66
SIN2	1767.06	688.88
STA	1933.07	760.81
RAND	458.99	120.79
EMP1	696.20	123.23
EMP2	201.08	36.72
EMP3	1054.01	316.17
EMP4	187.17	44.98
Fixed ordering cost		
500	1039.49	901.76
1000	844.54	583.64
1500	597.41	362.24
Penalty cost		
5	792.97	615.24
10	871.05	749.53
20	817.42	663.10
Coefficient of variation		
0.1	744.61	617.16
0.2	838.61	682.91
0.3	898.11	723.86
Average	827.15	679.02

Table 5: BS heuristics on a 25-period test bed, average computational times (seconds) with different setting parameters

to 597.41s as the fixed ordering cost increases from 500 to 1500. In contrast, with the increase of coefficient of variation, the computational times rise significantly. For instance, when the coefficient of variation rises from 0.1 to 0.2, the computational time increases from 744.61s to 838.61s. Whereas, standard deviations are large for all test instances. On average, the computational time is 827.15s and the standard deviation is 679.02s.

8. Conclusion

In this paper we discussed two MINLP-based heuristics for tackling non-stationary stochastic lot-sizing problems under (s, S) policy. These heuristics are based on mathematical programming models that can be solved by using off-the-shelf optimization packages. More specifically, we introduced the first MINLP model for computing near-optimal nonstationary (s, S) policy parameters and a binary search strategy to tackle larger-size problems. These MINLP models can be linearised via the approach discussed in Rossi et al. (2015) and can be implemented in OPL by adopting the `piecewise` expression.

We conducted an extensive computational study comprising 270 instances. We considered ten demand patterns, three fixed ordering costs, three penalty costs and three coefficients of variation.

For the 8-period numerical study, we investigated the performance of both models by contrasting costs of the policy obtained with our models against costs of the optimal policy obtained via the stochastic dynamic programming. Optimality gaps observed are generally below 0.3%. Our sensitivity analysis showed that the optimality gap is tighter when the demand keeps stable, and performance deteriorate with the increase of the penalty cost and the coefficient of variation; both models provide tighter gaps than those reported in the literature (Askin, 1981; Bollapragada and Morton, 1999).

The computational study carried out on larger instances (25-period planning horizon) showed that the computational efficiency of the binary search approach is reasonable: around 827.15s on average. Our sensitivity analysis demonstrates

that the computational time is positively correlated to the penalty cost and coefficient of demand variation, and has negative correlation with the fixed ordering cost.

References

Archibald, B.C., Silver, E.A., 1978. (s, S) policies under continuous review and discrete compound poisson demand. Management Science 24, 899–909. doi:10.1287/mnsc.24.9.899.

Askin, R.G., 1981. A procedure for production lot sizing with probabilistic dynamic demand. AIIE Transactions 13, 132–137. doi:10.1080/05695558108974545.

Bellman, R., 1957. Dynamic programming. Princeton University Press 89, 92.

Bollapragada, S., Morton, T.E., 1999. A simple heuristic for computing nonstationary (s, S) policies. Operations Research 47, 576–584. doi:10.1287/opre.47.4.576.

Bookbinder, J.H., Tan, J.Y., 1988. Strategies for the probabilistic lot-sizing problem with service-level constraints. Management Science 34, 1096–1108. doi:10.1287/mnsc.34.9.1096.

Dural-Selcuk, G., Kilic, O.A., Tarim, S.A., Rossi, R., 2016. A comparison of non-stationary stochastic lot-sizing strategies. arXiv:1607.08896 .

Federgruen, A., Zipkin, P., 1984. An efficient algorithm for computing optimal (s, S) policies. Operations research 32, 1268–1285. doi:10.1287/opre.32.6.1268.

Feng, Y., Xiao, B., 2000. A new algorithm for computing optimal (s, S) policies in a stochastic single item/location inventory system. IIE Transactions 32, 1081–1090. doi:10.1080/07408170008967463.

Graves, S.C., 1999. A single-item inventory model for a nonstationary demand process. Manufacturing & Service Operations Management 1, 50–61. doi:10.1287/msom.1.1.50.

IBM, 2011. IBM ILOG CPLEX Optimization Studio OPL Language Reference Manual.

Iglehart, D.L., 1963. Optimality of (s, S) policies in the infinite horizon dynamic inventory problem. Management science 9, 259–267. doi:10.1287/mnsc.9.2.259.

Rossi, R., Kilic, O.A., Tarim, S.A., 2015. Piecewise linear approximations for the static–dynamic uncertainty strategy in stochastic lot-sizing. Omega 50, 126–140. doi:10.1016/j.omega.2014.08.003.

Rossi, R., Tarim, S.A., Prestwich, S., Hnich, B., 2014. Piecewise linear lower and upper bounds for the standard normal first order loss function. Applied Mathematics and Computation 231, 489–502. doi:10.1016/j.amc.2014.01.019.

Sahin, I., 1982. On the objective function behavior in (s, S) inventory models. Operations Research 30, 709–724. doi:10.1287/opre.30.4.709.

Scarf, H.E., 1960. Optimality of (s, S) policies in the dynamic inventory problem, in: Arrow, K.J., Karlin, S., Suppes, P. (Eds.), Mathematical Methods in the Social Sciences. Stanford University Press, Stanford, CA, pp. 196–202.

Stidham Jr, S., 1977. Cost models for stochastic clearing systems. Operations Research 25, 100–127. doi:10.1287/opre.25.1.100.

Strijbosch, L.W., Syntetos, A.A., Boylan, J.E., Janssen, E., 2011. On the interaction between forecasting and stock control: the case of non-stationary demand. International Journal of Production Economics 133, 470–480. doi:10.1016/j.ijpe.2009.10.032.

Tarim, S.A., Kingsman, B.G., 2006. Modelling and computing (R^n, S^n) policies for inventory systems with non-stationary stochastic demand. European Journal of Operational Research 174, 581–599. doi:10.1016/j.ejor.2005.01.053.

Veinott Jr, A.F., Wagner, H.M., 1965. Computing optimal (s, S) inventory policies. Management Science 11, 525–552. doi:10.1287/mnsc.11.5.525.

Zheng, Y.S., Federgruen, A., 1991. Finding optimal (s, S) policies is about as simple as evaluating a single policy. Operations research 39, 654–665. doi:10.1287/opre.39.4.654.

Appendix A. The `piecewise` OPL constraint

Rossi et al. (2015) piecewise linearised loss functions in constraints (8) and (9) by employing piecewise linear approximations based on Jesen's and Edmundson-Madanski inequalities. An alternative strategy is to model these non-linear functions by exploring the `piecewise` syntax in OPL. By using this syntax, a piecewise function is specified by giving a set of slopes which represent the linear variation for each linear segment; a set of breakpoints at which slopes change; and the function value at a known point.

```
piecewise(i in 1..W){
slope[i] -> breakpoint[i];
 slope[W+1]
}(<knownpoint>,<valuepoint>)<value>;
```

Figure A.6: The syntax of the `piecewise` command in OPL

The `piecewise` syntax in OPL is given in Figure A.6. `W` is the number of breakpoints of the piecewise function. `slope[i]` and `breakpoint[i]` denote slope and breakpoint of segment i. Segment i goes from breakpoint $(i-1)$ to breakpoint (i). `<valuepoint>` is the function value at a known point

`<knownpoint>`. Finally, `<value>` represents the value at which we evaluate the function.

For the OPL `piecewise` syntax, there are three key components: slope, breakpoint, and function value at a known point. The following lemmas will demonstrate how to deduce their values. Let Ω be the support of ω. Let $(\Omega_i)_{i=1,\dots,W+1}$ be a partition of Ω in $W+1$ segments.

Lemma 1. *The slope of i^{th} segment is written as*

$$l_i = \sum_{k=1}^{i-1} p_k, i \in \{1, 2, \dots, W+1\},$$

where $p_i = Pr\{\omega \in \Omega_i\} = \int_{\Omega_i} g_\omega(t)dt$, $g_\omega(\cdot)$ denotes the probability density function of ω.

Proof 1. *Observation from Rossi et al. (2014), Lemma 11.*

Lemma 2. *The i^{th} breakpoint can be written as*

$$X_i = E[\omega|\Omega_i], i \in \{1, 2, \dots, W\}.$$

Proof 2. *Observation from Rossi et al. (2014), Lemma 11.*

Note that when ω follows a normal distribution with mean μ and standard deviation σ, then $\hat{L}_{\text{up}}(x, \omega) = \sigma \hat{L}_{\text{up}}(\frac{x-\mu}{\sigma}, Z)$, where Z follows a standard normal distribution, see Lemma 7 in Rossi et al. (2014).

Lemma 3. *Assume that the partition of Ω is symmetric with respect to 0, then the function value $\hat{L}_{up}(x, \omega)$ at point 0 can be written as follows.*

$$\hat{L}_{up}(0, \omega) = \begin{cases} -\sum_{k=1}^{\frac{W+1}{2}} p_k E[\omega|\Omega_k] + e_W, & W \text{ is odd} \\ -\frac{1}{2}(\sum_{k=1}^{\frac{W}{2}} p_k E[\omega|\Omega_k] + \sum_{k=1}^{\frac{W}{2}+1} p_k E[\omega|\Omega_k]) + e_W, & W \text{ is even} \end{cases}$$

where e_W represents the approximation error.

Proof 3. *Since the partition of Ω is symmetric when W is odd, $x = 0$ is the central breakpoint. Hence, the function value at this breakpoint can be calculated directly. However, when W is even, the function value at point $x = 0$ is the average of nearest two symmetric breakpoints $X_{\frac{W}{2}}$ and $X_{\frac{W}{2}+1}$.*

Following Lemma 1, 2 and 3, constraint (8) and (9) in Fig. 2 can be expressed as Eq. (A.1) and (A.2) in Fig. A.7, for $t = 1, \ldots, T$.

$$P_{jt} = 1 \rightarrow H_t = \texttt{piecewise}\{l_i \rightarrow X_i; 1\}(0, \hat{L}_{up}(0, d_{jt}))\tilde{I}_t, \quad i = 1, \ldots, W;\ j = 1, \ldots, t. \quad \text{(A.1)}$$

$$P_{jt} = 1 \rightarrow B_t = \texttt{piecewise}\{-1 + l_i \rightarrow X_i; 0\}(0, \hat{L}_{up}(0, d_{jt}))\tilde{I}_t, \quad i = 1, \ldots, W;\ j = 1, \ldots, t. \quad \text{(A.2)}$$

Figure A.7: Rewriting holding and penalty costs by adopting `piecewise` syntax

Appendix B. The MILP model

The joint MILP model to calculate near-optimal (s, S) policy parameters for the non-stationary stochastic lot-sizing problem is presented below. Note that we plug in the original fomulations (B.9), (B.10), (B.21), and (B.22) to our joint MILP model in order to enhance the computational perforance without excessively compromising solution quality.

$$\begin{aligned}\min \Big(& -cI_0^S + c\sum_{t=1}^{T}\tilde{d}_t^S + \sum_{t=1}^{T}(K\delta_t^S + hH_t^S + bB_t^S) + c\tilde{I}_T^S \\ & -cI_0^s + c\sum_{t=1}^{T}\tilde{d}_t^s + \sum_{t=2}^{T}(K\delta_t^s + hH_t^s + bB_t^s) + c\tilde{I}_T^s\Big) \end{aligned} \quad \text{(B.1)}$$

Subject to, $t = 1, \ldots, T$

$$C_t^S(I_0^S) = -cI_0^S + c\sum_{t=1}^{T}\tilde{d}_t + \sum_{t=1}^{T}(K\delta_t^S + hH_t^S + bB_t^S) + c\tilde{I}_T^S \quad \text{(B.2)}$$

$$\tilde{I}_t^S + \tilde{d}_t - \tilde{I}_{t-1}^S \geq 0 \quad \text{(B.3)}$$

$$\tilde{I}_t^S + \tilde{d}_t - \tilde{I}_{t-1}^S \leq \delta_t^S M \quad \text{(B.4)}$$

$$\sum_{j=1}^{t} P_{jt}^S = 1 \quad \text{(B.5)}$$

$$P_{jt}^S \geq \delta_j^S - \sum_{k=j+1}^{t}\delta_k, j = 1, \ldots, t \quad \text{(B.6)}$$

$$\delta_1^S = 1 \tag{B.7}$$

$$I_0^S = \tilde{I}_1^S + \tilde{d}_1 \tag{B.8}$$

$$H_t^S \geq (I_t^S + \sum_{j=1}^{t} d_{jt} P_{jt}^S) \sum_{k=1}^{i} p_k - \sum_{j=1}^{t} (\sum_{k=1}^{i} p_k E[d_{jt}|\Omega_i] - e_W) P_{jt}^S, \qquad i = 1, \cdots, W \tag{B.9}$$

$$B_t^S \geq -I_t^S + (I_t^S + \sum_{j=1}^{t} d_{jt} P_{jt}^S) \sum_{j=1}^{i} p_k - \sum_{j=1}^{t} (\sum_{k=1}^{i} p_k E[d_{jt}|\Omega_i] - e_W) P_{jt}^S, \qquad i = 1, \cdots, W \tag{B.10}$$

$$P_{jt}^S \in \{0, 1\}, j = 1, \ldots, t \tag{B.11}$$

$$\delta_t^S \in \{0, 1\} \tag{B.12}$$

$$G_t^s(I_0^s) = -cI_0^s + (hH_1^s + bB_1^s) + c\sum_{t=1}^{T} \tilde{d}_t + \sum_{t=2}^{T} (K\delta_t^s + hH_t^s + bB_t^s) + c\tilde{I}_T^s \tag{B.13}$$

$$\tilde{I}_t^s + \tilde{d}_t - \tilde{I}_{t-1}^s \geq 0 \tag{B.14}$$

$$\tilde{I}_t^s + \tilde{d}_t - \tilde{I}_{t-1}^s \leq \delta_t^s M \tag{B.15}$$

$$\sum_{j=1}^{t} P_{jt}^s = 1 \tag{B.16}$$

$$P_{jt}^s \geq \delta_j - \sum_{k=j+1}^{t} \delta_k^s, j = 1, \ldots, t \tag{B.17}$$

$$\delta_1^s = 0 \tag{B.18}$$

$$P_{jt}^s = 1 \rightarrow H_t^s = \texttt{piecewise}\{l_i \rightarrow X_i; 1\}(0, \hat{L}_{up}(0, d_{jt}))\tilde{I}_t^s, \qquad \begin{matrix} i = 1, \ldots, W \\ j = 1, \ldots, t \end{matrix} \tag{B.19}$$

$$P_{jt}^s = 1 \rightarrow B_t^s = \texttt{piecewise}\{-1 + l_i \rightarrow X_i; 0\}(0, \hat{L}_{up}(0, d_{jt}))\tilde{I}_t^s \qquad \begin{matrix} i = 1, \ldots, W \\ j = 1, \ldots, t \end{matrix} \tag{B.20}$$

$$H_t^s \geq (I_t^s + \sum_{j=1}^{t} d_{jt} P_{jt}^s) \sum_{k=1}^{i} p_k - \sum_{j=1}^{t} (\sum_{k=1}^{i} p_k E[d_{jt}|\Omega_i] - e_W) P_{jt}^s, \qquad i = 1, \cdots, W \tag{B.21}$$

$$B_t^s \geq -I_t^s + (I_t^s + \sum_{j=1}^{t} d_{jt} P_{jt}^s) \sum_{j=1}^{i} p_k - \sum_{j=1}^{t} (\sum_{k=1}^{i} p_k E[d_{jt}|\Omega_i] - e_W) P_{jt}^s, \qquad i = 1, \cdots, W \tag{B.22}$$

$$P_{jt}^s \in \{0, 1\}, j = 1, \ldots, t \tag{B.23}$$

$$\delta_t^s \in \{0, 1\} \tag{B.24}$$

$$I_0^s \leq \tilde{I}_1^S + \tilde{d}_1 \tag{B.25}$$

$$G_1^s(I_0^s) = C_1^S(\tilde{I}_1^S + \tilde{d}_1) \tag{B.26}$$

Appendix C. Test bed

Periodic demands with different demand patterns under the eight period computational study are displayed in Table C.6. The demand of each period under the twenty-five periods numerical example is shown in Table C.7. The first column represents period indexes; the rest columns denote various demands.

Period	LCY1	LCY2	SIN1	SIN2	STA	RAND	EMP1	EMP2	EMP3	EMP4
1	15	3	15	12	10	2	5	4	11	18
2	16	6	4	7	10	4	15	23	14	6
3	15	7	4	7	10	7	26	28	7	22
4	14	11	10	10	10	3	44	50	11	22
5	11	14	18	13	10	10	24	39	16	51
6	7	15	4	7	10	10	15	26	31	54
7	6	16	4	7	10	3	22	19	11	22
8	3	15	10	12	10	3	10	32	48	21

Table C.6: Demand data of the 8-period computational analysis

Period	LCY1	LCY2	SIN1	SIN2	STA	RAND	EMP1	EMP2	EMP3	EMP4
1	11	23	130	122	100	178	2	47	44	49
2	17	32	150	130	100	178	51	81	116	188
3	26	42	127	120	100	136	152	236	264	64
4	38	55	76	98	100	211	467	394	144	279
5	53	70	27	77	100	119	268	164	146	453
6	71	86	10	70	100	165	489	287	198	224
7	92	103	36	81	100	47	446	508	74	223
8	115	120	88	103	100	100	248	391	183	517
9	138	136	136	124	100	62	281	754	204	291
10	159	150	149	130	100	31	363	694	114	547
11	175	161	121	118	100	43	155	261	165	646
12	186	168	68	95	100	199	293	195	318	224
13	190	170	22	75	100	172	220	320	119	215
14	186	168	11	71	100	96	93	111	482	440
15	175	161	42	84	100	69	107	191	534	116
16	159	150	96	107	100	8	234	160	136	185
17	138	136	140	126	100	29	124	55	260	211
18	115	120	148	129	100	135	184	84	299	26
19	92	103	114	115	100	97	223	58	76	55
20	71	86	60	91	100	70	101	0	218	0
21	53	70	18	73	100	248	123	0	323	0
22	38	55	14	72	100	57	99	0	102	0
23	26	42	50	87	100	11	31	0	174	0
24	17	32	104	110	100	94	82	0	284	0
25	11	23	144	127	100	13	0	0	0	0

Table C.7: Demand data of the 25-period computational analysis