On computational procedures for Value Iteration in inventory control [⋆]

Eligius M.T. Hendrix [*] **Cleo Kortenhorst** [**] **Gloria L. Ortega** [*]

[*] *Computer Architecture, Universidad de Málaga (e-mail: {eligius,gloriaortega@uma.es)*
[**] *Operations Research and Logistics, Wageningen University (e-mail: cleo.kortenhorst@wur.nl).*

Abstract: Dynamic programming (DP) is often seen in inventory control to lead to optimal ordering policies. When considering stationary demand, Value Iteration (VI) may be used to derive the best policy. In this paper, our focus is on the computational procedures to implement VI. Practical implementation requires bounding carefully the state space and demand in an adequate way. We illustrate with small cases the challenge of the implementation. We also show that handling service level constraints is not straightforward from a DP point of view. Moreover, when taking the age distribution into account for perishable products, the curse of dimensionality provides an additional challenge. Copyright © 2019 IFAC

Keywords: Mathematical Approaches for Scheduling, Stochastic Processes, Inventory control, Value Iteration, Service level, Poisson

1. INTRODUCTION

As inventory control is a dynamic process, also dynamic programming has been considered an appropriate technique to derive so-called order policies, see Silver et al. (1998). We use here small retailer models to illustrate the determination of the optimal order policy. In many cases, a so-called base stock policy works well, where an order-up-to level S is used, Minner (2000). The retailer inspects the current inventory level I and orders an amount $Q = q(I)$ up to level S, i.e. the rule $q(I) = (S-I)^+$ is followed, where $(x)^+ = \max\{x, 0\}$. However, whether this is the optimal policy $q(I)$ depends on the drive towards ordering.

For some inventory models, it is possible to derive the so-called optimal order policy via Stochastic Dynamic Programming (SDP) using a computational procedure which is called Value Iteration (VI). This paper discusses the features of deriving the optimal policy implementing Value Iteration using several small inventory control situations. In all situations, we consider a fixed order cost k and inventory holding cost h. As such this would lead to ordering nothing at all, so there are different drives in the practical situation. We fist consider a case with backordering. The second (and fourth) situation takes sales into account and the third situation considers a service level requirement.

This paper is organised as follows. We first sketch the concept of deriving the optimal policy by VI in Section 2. Section 3 then introduces and investigates several cases based on Poisson distributed demand. Section 4 summarizes our findings.

[⋆] This work has been partially funded by grant RTI2018-095993 from the Spanish Ministry in part financed by the European Regional Development Fund (ERDF).

2. THE PROCEDURE OF VI

First of all, to define the ordering decision problem as a Markov Decision Problem (MDP), we have to define the state space, decision space and transition probabilities. For the inventory situation, this requires to have a good vision on the order of events. For our retailer case, we assume that the order of size $Q_t = q(I_t)$ is placed and received next morning, just before the next decision is made based on the total inventory I_t in stock. The dynamics is following the equation

$$I_{t+1} = I_t + q(I_t) - \mathbf{d}_t, \tag{1}$$

where $\mathbf{d}_t$ is the stochastic customer demand during day t. Notice that we implicitly think in time steps of one day instead of in terms of continuous time, which is also reasonable for a retailer situation. The state space $\mathcal{S}$ translates here easily to the inventory I in stock. However, in most MDP situations, this is thought of as a finite state space. Apparently, we also should think of a discretized space here. For an overview of practical problems, we refer to the book Boucherie and van Dijk (2017). We will first look at the theoretical background to embed the inventory problem and then derive the VI procedure to be followed.

2.1 MDP background

Under certain circumstances, the optimal order amount $q^*(I)$ can be characterized by the MDP theory, introduced originally by Bellman (1957). For the optimal policy, there exists a so-called value function $v(I)$ and a scalar π such that $\forall I \in \mathcal{S}$

$$v(I) + \pi = \min_Q \left(C(I,Q) + \int_0^\infty v(I+Q-d)p_d \right), \tag{2}$$

where p_d is the probability density of demand $\mathbf{d}$ and $C(I,Q)$ the costs associated with inventory I and order quantity Q.

To work with (2) in a practical way considering $\mathcal{S}$ to be finite, we observe two problems. If $\mathbf{d}$ is from a continuous distribution, we require what is called *interpolation* to evaluate v in between known grid points. Moreover, if the support of $\mathbf{d}$ is not finite, we may arrive at state values out of range and require *extrapolation* to evaluate v outside of the range. An example of such a system in fishery dynamics is given in van Dijk et al. (2014).

The practical cases we are interested in concern low discrete demand of perishable items, see Ortega et al. (2018), which may lead to hard to determine optimal policies. This means, in the sequel, we consider $\mathbf{d}$ from a discrete distribution with low mean and $\mathcal{S} \subset \mathbb{Z}$. This translates the optimality condition (2) to the existence of a vector V and constant π such that

$$V(I) + \pi = \min_{Q} \left(C(I,Q) + \sum_{d=0}^{\infty} p_d V(I+Q-d) \right), \quad (3)$$

where now p_d is the probability mass function of $\mathbf{d}$. The optimal decision rule (called order policy) is given by

$$q^*(I) = \mathrm{argmin}_Q \left(C(I,Q) + \sum_{d=0}^{\infty} p_d V(I+Q-d) \right). \quad (4)$$

This is the theory, but how to derive the optimal policy now in practice? And, do we always reach an optimal policy?

2.2 Value Iteration procedure

In principle, the optimal valuation can be found following a fixed point idea about (3) with respect to value π called Value Iteration (VI). Although terminology and concepts are more extensive, from a computational point of view it is sufficient to think in those terms. One copies the last valuation in a vector W and determines the new valuation vector V as sketched in Algorithm 1 up to convergence takes place of what is called the span, i.e. the maximum and minimum difference of the two vectors.

Algorithm 1. Pseudocode of VI for one product inventory control

```
1: Set vector elements $V_i$ to $C(i,0)$ for $i = 0,\ldots,N-1$
2: repeat
3:     Copy vector $V$ into vector $W$
4:     for $i = 0,\ldots,N-1$ do
5:         for $Q = 0,\ldots,\overline{Q}$ do
6:             for all demand realisations (events) $d$ do
7:                 Retrieve $W_j$, with state $j = i+Q-d$
8:         $V_j = \min_Q[C(i,Q) + \sum_d p_d W_j]$
9: until $\max_j(V_j - W_j) - \min_j(V_j - W_j) < \epsilon$
```

As mentioned before, we might in line 7 arrive out of range when demand is larger than the available material. There are two solutions. Either we follow a lost sales model and the next inventory state is given by $j = (i-d)^+ + Q$, or we follow a backlogging model that allows negative inventory positions. The latter requires to define a minimum value $\underline{I}$ for the inventory position and to apply extrapolation. Bounding plays a big role in the implementation of VI. In the case we sketch, we also should define a maximum order quantity $\overline{Q}$. Moreover, the range of inventory values $\mathcal{S} = [\underline{I}, \overline{I}]$ should be realistic in the sense that all positions have a positive probability to be reached. Extrapolation for values larger than $\overline{I}$ can be avoided limiting the order quantities to be evaluated to $Q \leq \max\{\overline{Q}, \overline{I} - I\}$. Figure 1 sketches the lost sales situation, where the inventory value I and state number i coincide.

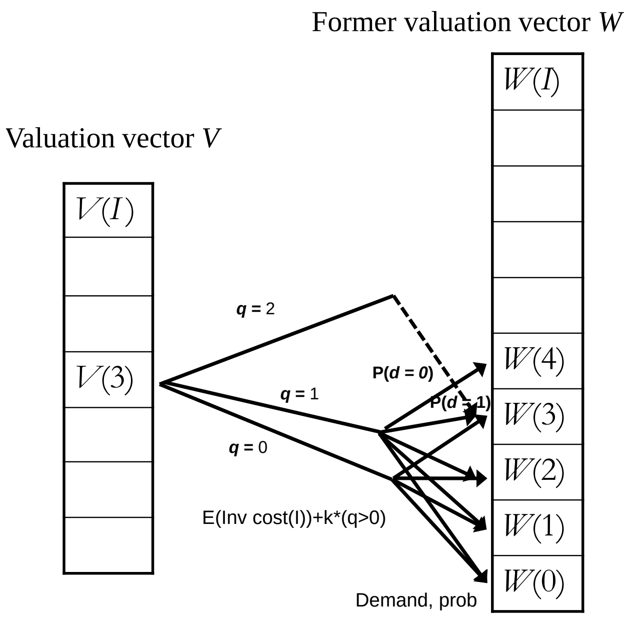

Fig. 1. Data dependence in Value Iteration

3. CASES

We consider a retailer case. Dynamics (1) represents the case of backlogging (with negative inventory) where a customer is prepared to wait for the order to be delivered. However, a retailer case mainly concerns a lost sales situation, where the demand (and potentially client) is lost, described by the dynamics

$$I_{t+1} = (I_t - \mathbf{d}_t)^+ + q(I_t). \quad (5)$$

This means that negative inventory does not occur. We will describe both situations here to focus on the consequences for the implementation of VI.

Notice that the expected inventory cost given a starting inventory I is due to the final inventory at the end of the day

$$E(Invcost(I)) = hE(I - \mathbf{d})^+. \quad (6)$$

For the illustration, we consider a Poisson demand with mean $\mu = 2$ and the cost values $k = 4$ and $h = 0.25$. In case of deterministic demand, this would lead to an economic order quantity $EOQ = \sqrt{\frac{2k\mu}{h}} = 8$ or alternatively, a so-called replenishment period of 4 days, see Silver et al. (1998). As ordering also implies cost, the optimal solution is to order zero if there is no drive for selling the product. We discuss low dimensional inventory cases with three different drives, i.e. that of backlogging, of having sales and that where the sales are forced by a so-called service level requirement.

3.1 First case, backlogging

We first introduce a backlogging cost of $c = 1$ per unit per day. This implies that I can take negative values and adds an expected cost $c(-I)^+$ to (6). How negative can $\underline{I}$ be? Let G be the cumulative distribution function (CDF) of $\mathbf{d}$. We consider a maximum demand as the .9999 percentile[1] of $\mathbf{d}$ giving $\overline{d} = G^{-1}(.9999) = 9$.

To determine the maximum inventory level $\overline{I}$ we take the EOQ plus a margin s reasoning that the retailer prefers to incur the inventory cost h over the backlogging cost c in $\frac{c-h}{c}$ cases, i.e. $s = G^{-1}(\frac{c-h}{c}) = 3$. This means we can take $\overline{I} = EOQ + s = 11$. For the minimum inventory level we reason that the retailer intends to stay over s in its final inventory level and may be confronted with the maximum demand $\underline{I} = s - \overline{d} = -6$, such that $\mathcal{S} = [\underline{I}, \overline{I}] = [-6, 11]$. As we are evaluating the VI for state values $[\underline{I}, \overline{I}] = [-6, 11]$, we still have to extrapolate the value of W for state values lower than $\underline{I}$.

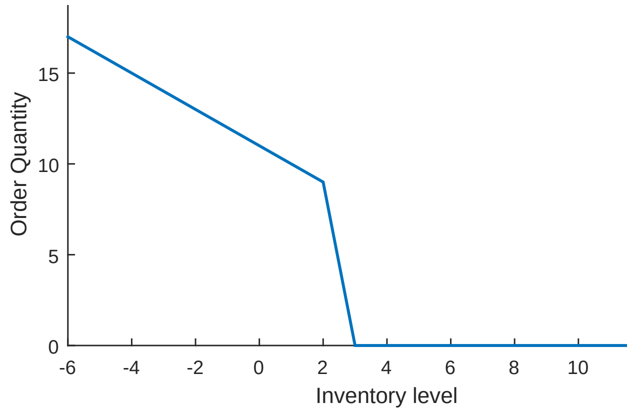

Fig. 2. Optimal order quantity $q(I)$ as function of the inventory level for case 1

Running the VI algorithm with an accuracy of $\epsilon = 10^{-5}$, provides the optimal solution as sketched in Figure 2 after 28 iterations to an average daily cost of $\pi = 2.01$. The outcome corresponds to a so-called (s, S) policy, where the retailer orders when the level drops below re-order point (value) $s = 3$ a quantity $Q = S - I$, with in this case order-up-to level $S = 11$. Such cases are typical in inventory theory (Minner (2000)), although it is also used in continuous review, i.e. the order takes place any moment instead of at the beginning of the day.

3.2 Second case, sales

Although popular in theory due to some nice mathematical properties, backlogging is not very realistic in the practice of retail situations of supermarkets. We consider here that the retailer is obtaining a profit of $f = 2$ per unit which is in fact lost on lost sales (demand is larger than inventory). In the bounding of the state space now $\underline{I} = 0$, such that extrapolation is not necessary anymore. For the maximum inventory $\overline{I}$ we take the maximum demand during the replenishment cycle in terms of the .9999 percentile, providing $\overline{I} = 20$, such that $\mathcal{S} = [\underline{I}, \overline{I}] = [0, 20]$.

We change the objective in maximising profit, where for each state I the profit is set to

$$C(I, q) = fE\min\{I, \mathbf{d}\} - hE(I - \mathbf{d})^+ - k(q > 0). \quad (7)$$

[1] Notice that the inverse $G^{-1}(\alpha) = x$ is in fact the smallest value x for which $G(x) \geq \alpha$, also called the percentile point.

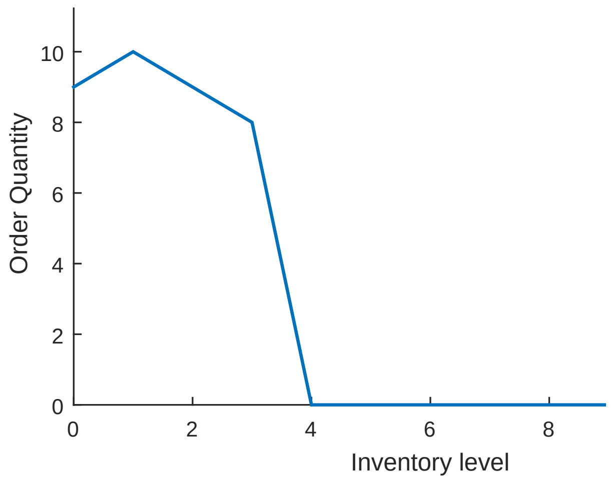

Fig. 3. Optimal order quantity $q(I)$ as function of the inventory level for case 2

Running the VI algorithm with an accuracy of $\epsilon = 10^{-5}$ provides now an average daily profit of $\pi = 1.77$. The optimal policy depicted in Figure 3 is surprising, as for $I = 0$, it deviates from an (s, S) policy. Of course the policy is data dependent. For lower values of the profit f, the policy first converges to a preemptive policy only ordering when being out of stock to the value of $f = 0.95$ under which the business is not profitable anymore and costs exceed expected profit.

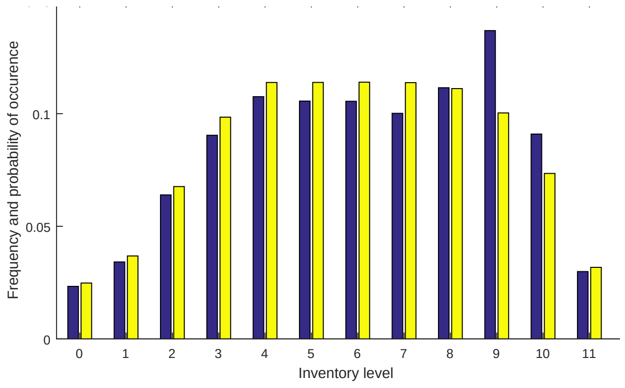

Fig. 4. Distribution inventory, frequency in blue and Markovian stationary state in yellow

Following a dynamic model with a finite number of states, allows us in this case to observe the occurrence of the state values. On one hand, we can simulate the model and count which inventory levels occur, but we can now also follow the theory of MDP and construct the Markov transition matrix M where each element M_{ij} represents the probability given the optimal order quantity $q(i)$ that we will reach state j. The so-called stationary state distribution of inventory levels is a row eigenvector of M belonging to eigenvalue 1, e.g. see Puterman (1994). Notice first of all, that the actual state space $\mathcal{S}$ is too large, as the maximum possible amount in stock is $\overline{I} = \max_i q(i) + \max(\text{argmax}_i\, q(i)) = 10 + 1 = 11$.

We can also observe this from the distribution in Figure 4. Notice for instance that the probability of being out of stock at the beginning of the day is extremely low. Given the optimal policy (Figure 3), this would imply having ordered nothing the day before, i.e. this can only occur if the inventory is $I \geq 4$ and the demand has been larger than I.

3.3 Third case, service level requirements

An often applied measure in inventory control is the use of service level requirements. The so called α-service level measures the number of days that the shop is out of stock before the end of the day and there is demand that cannot be fulfilled. To be more precise, in our model, a requirement could be

$$P(\mathbf{d}_t \leq I_t) \geq \alpha, \tag{8}$$

where for instance $\alpha = 0.9, 0.95, 0.97$. As illustrated in Pauls-Worm and Hendrix (2015) for a case of non-stationary demand, this is hard to capture in an optimal way in dynamic programming. What can be implemented in a DP approach is to have a conditional probability, for the current inventory level I, that at the end of the next day the requirement is fulfilled. However, as depicted in Figure 4, not each inventory level I has the same probability of occurence causing DP to over achieve the requirement (8). I.e. for some states (inventory levels) one could relax the conditional constraint. However, for which levels is only known after achieving a certain order policy.

Moreover, the conditional constraints also have to be derived with care. For the situation $i = 0$, we know we should order $q(0) \geq s_0(\alpha) = G^{-1}(\alpha)$ to be certain that next day (this day is lost), we have α probability to have enough in stock. For an inventory level $i \geq 1$ we derive the minimum order quantity $s_i(\alpha)$ in a conditional way on the inventory level j at the end of the day. Let $s_i(\alpha)$ be the minimum quantity s for which

$$\sum_{j=0}^{i} P(\mathbf{d}_{t+1} \leq s + j) \times P((i - \mathbf{d}_t)^+ = j) \geq \alpha. \tag{9}$$

We can first derive the conditional probabilities of the so-called loss function $\Pi_{ij} = P((i - \mathbf{d})^+ = j|i)$ and then solve (9) over s using the equation

$$\sum_{j=0}^{i} \Pi_{ij} G(s_i(\alpha) + j) \geq \alpha. \tag{10}$$

Let F be the CDF of the demand $\mathbf{d}_t + \mathbf{d}_{t+1}$, which in case of Poisson demand is also Poisson distributed with mean 2μ. Then we have that for $i \geq F^{-1}$ we can take $s_i(\alpha)$=0, as there is sufficient material in stock for the next two days.

The conditional requirement can be taken as $q \in [s_i(\alpha), \max\{\overline{Q}, \overline{I} - I\}]$ as action space to prevent "ordering nothing" to be the optimal solution. As maximum inventory level we can take the EOQ again plus the margin that intends to have enough for the coming two days; $\overline{I} = EOQ + F^{-1}(\alpha)$.

Using a service level requirement of $\alpha = 90\%$ and the accuracy of $\epsilon = 10^{-5}$, the value iteration converges in 20 iterations to an average daily cost of $\pi = 2.61$. The shape of the order policy provides the same surprise, that it does not completely follow an (s, S) policy. The α-service level reaches a level of 0.99, i.e. much higher than the required amount. In this specific case, this is caused by

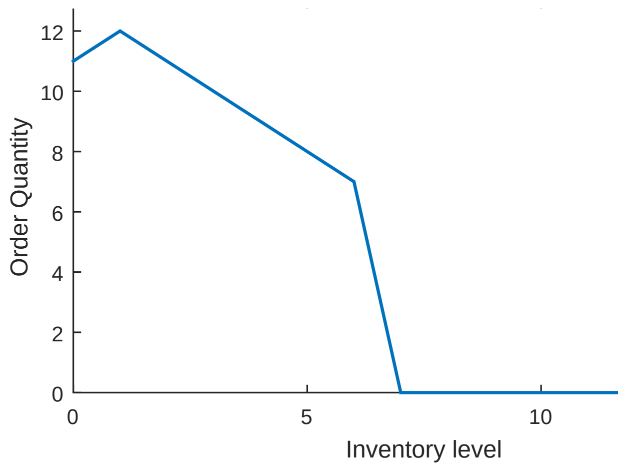

Fig. 5. Optimal order quantity $q(I)$ for case 3

the relatively large replenishment cycle length, such that most of the days, the inventory cannot be out of stock.

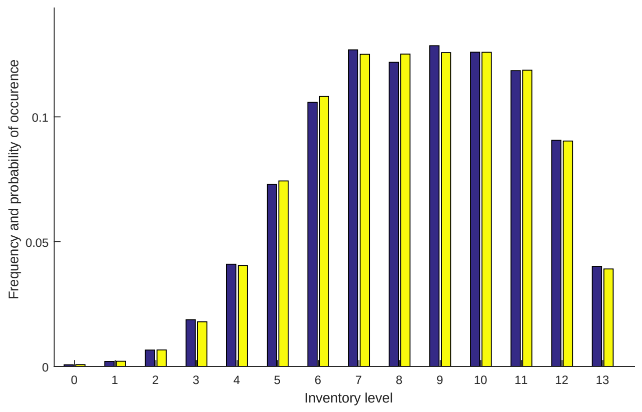

Fig. 6. Distribution inventory case 3, frequency in blue and Markovian stationary state in yellow

This specific case with a long replenishment cycle and low demand illustrates very well that the Dynamic Programming approach with an individual requirement $q \geq s_i(\alpha)$ for each state i according to (10) exceeds the global requirement (9) and does not necessarily lead to an optimal solution. Running the same VI with an individual requirement of $\alpha = 0.6$ provides a global service level of 0.91 against an average daily cost of $\pi = 1.89$ according to a simulation over 200,000 days.

3.4 Fourth case, perishable products

This last case illustrates how taking the shelf life of products into account, not only the state space dimension is increasing, but also the underlying MDP theory may be violated due to the parameter values. Consider again the second case where the retailer sells products against a price $s = 3$ and buys for a price $c = 1$ such that the profit is the same. However, now the product is highly perishable and expires after two days after delivery. The state space becomes two-dimensional taking the age of the product into account; I_0 is the fresh product delivered in the morning and I_1 is already a day old and will expire wen not sold at the end of the day. We consider a so-called

FIFO issuing policy, where the client first buys the oldest product and then the fresher one. The dynamics of the model is now

$$I_{0t+1} = Q_t \tag{11}$$

and

$$I_{1t+1} = (I_{0t} - (d_t - I_{1t})^+)^+. \tag{12}$$

The objective in terms of random variables is now

$$C(I_0, I_1, q) = sE\min\{I_0 + I_1, \mathbf{d}\} - hI_1 - cq - k(q > 0). \tag{13}$$

The generated waste is in fact given by $(I_{1t} - d_t)^+$ and is implcit in the model as we loose the buying price c of the product. Having a maximum order quantity $\overline{Q}$, we can now limit the state space to the two dimensional space $(I_0, I_1) \in \mathcal{S} := \{0, .., \overline{Q}\}^2$. Using the same parameter

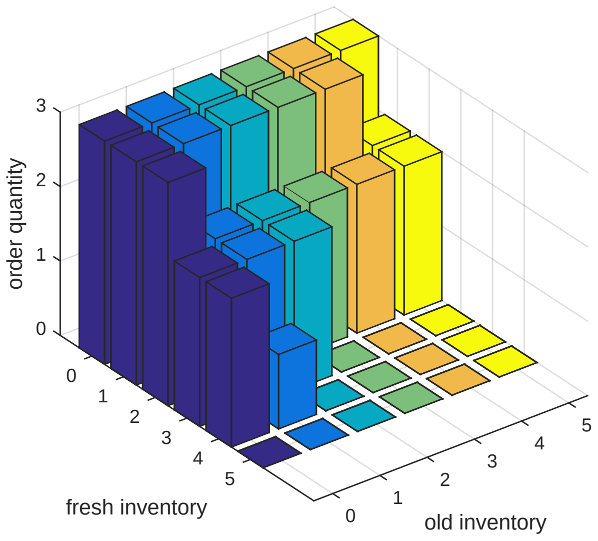

Fig. 7. Optimal order quantity $q(I_0, I_1)$ case 4, for parameter values $s = 3, c = 1, h = 0.1, k = 0$

values as in case 2, $k = 4, h = 0.25$, leads to an optimal replenishment cycle of 4 periods, which is not relevant anymore due to the shelf life of 2 days of the product. Interesting and illustrative is that for such a case, the optimal policy provides a periodic policy which means to order every two periods 5 units. One period an order is placed and the next one nothing is ordered, such that all items in stock expire. In such a case, the value iteration per period is not valid anymore and we cannot follow the procedure we were following before.

To force the policy to replenish also when not being out of stock, we removed the order cost setting $k = 0$ and lowered the inventory cost $h = 0.1$ using a maximum order quantity of $\overline{Q} = 5$. Running the value iteration provides an optimal policy after 12 iterations to an average daily profit of $\pi = 3.15$ as depicted in Figure 7.

Like in the former cases, we can analyse the behavior of the system from a Markovian perspective. First notice that the optimal policy has been derived for $(I_0, I_1) \in \{0, .., 5\}^2$. However, the maximum inventory for each age is in fact $\max_{I_0 I_1} q(I_0, I_1)$ which is $\overline{Q} = 3$ in the given numerical case. This means, there are many transient states where the Markov chain is not returning for which we calculated the optimal policy.

Computing the stationary state which presents the probability distribution of being in a certain state, requires computing the transient probabilities where we have to map the two-dimensional state space into a one-dimensional one to define the Markov matrix M. The stationary state

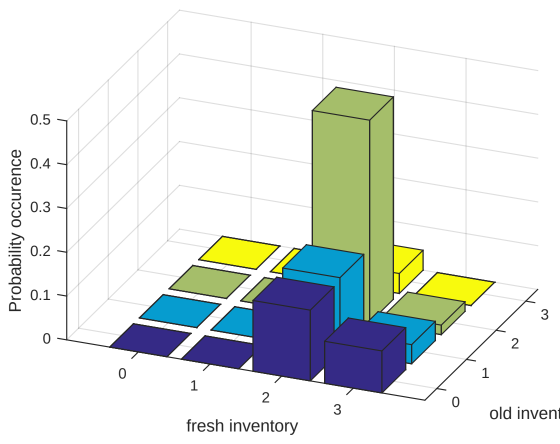

Fig. 8. Probability of occurence of inventory states for case 4 of a perishable product

according to the eigenvalue of the Markov chain coincides with the frequency measured by the simulation model and is depicted in Figure 8. Notice that as soon we leave the zero inventory state, we never return there and the fresh inventory fluctuates among $I_0 = 2, 3$ as in every period an order of this size is placed.

4. CONCLUSIONS

This contribution illustrated with small one product examples several characteristics of finding the optimal order policy using value iteration in an MDP model. Working with a discrete demand distribution facilitates to work over a grid of inventory values in state space. However, bounding of demand is of importance for a backlogging situation as minimum inventory levels may be negative. Secondly, bounding of state space and action space is of importance to let the method converge. In backlogging situations, we also have to take care of extrapolation of the value function.

Although the use of service level constraints is popular in inventory control, there is a challenge to find the optimal solution, as only conditional requirements can be developed as a minimum order quantity for each state individually, independently of the occurence of the state. For small instances as we used, the stationary distribution can also be derived using the row eigenvector of the Markov matrix.

This paper also illustrated the idea o a multi-dimensional state space when taking the age distribution into account for perishable products with a finite shelf life, which basically may lead to the curse of dimensionality. We observed there that in some situations the system may start to behave in a periodic way, such that the Value Iteration should be adapted to the periodic behavior.

REFERENCES

Bellman, R. (1957). A Markovian decision process. *Journal of Mathematics and Mechanics*, 6(5), 679–684.

Boucherie, R.J. and van Dijk, N.M. (2017). *Markov Decision Processes in Practice.* Springer International Publishing.

Minner, S. (2000). *Strategic Safety Stocks in Supply Chains.* Springer.

Ortega, G., Hendrix, E.M.T., and García, I. (2018). A CUDA approach to compute perishable inventory control policies using value iteration. *The Journal of Supercomputing.*

Pauls-Worm, K.G.J. and Hendrix, E.M.T. (2015). SDP in inventory control: Non-stationary demand and service level constraints. In O. Gervasi et al. (eds.), *Computational Science and Its Applications – ICCSA 2015*, 397–412. Springer, Cham.

Puterman, M.L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, Inc., New York, NY, USA, 1st edition.

Silver, E., Pyke, D., and Peterson, R. (1998). *Inventory Management and Production Planning and Scheduling.* Wiley.

van Dijk, D., Hendrix, E.M.T., Haijema, R., Groeneveld, R.A., and van Ierland, E.C. (2014). On solving a bi-level stochastic dynamic programming model for analyzing fisheries policies: Fishermen behavior and optimal fish quota. *Ecological Modelling*, 272, 68 – 75.