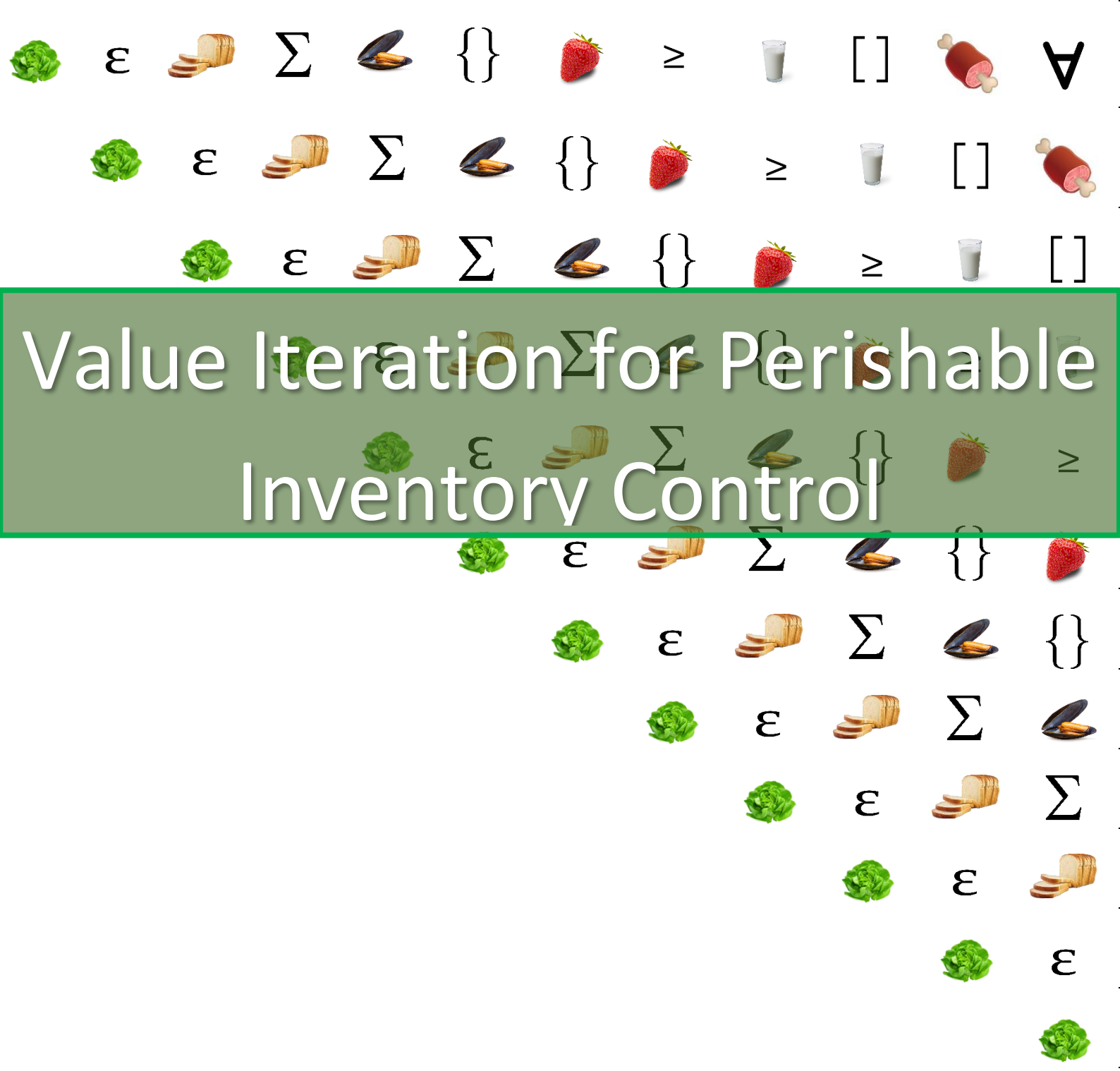
Value Iteration for Perishable Inventory Control

Author: Cleo G.F. Kortenhorst
MSc Research report
Universidad de Málaga in co-operation with Wageningen University
Computer Architecture and Operations Research and Logistics

Grant RTI2018-095993 from the Spanish state
in part financed by the European Regional Development Fund (ERDF)
Supervisors: Dr. E.M.T. Hendrix and Dr.K.G.J. Pauls-Worm

May 13, 2019



Value Iteration for Perishable Inventory Control

Cleo G.F. Kortenhorst
Studentnumber: 940916469090

29-3-2019

Supervisors:
Eligius M.T. Hendrix (WUR)
Karin G.J. Pauls-Worm (WUR)

MSc Thesis
Operations Research & Logistics
MSc Management, Economics and Consumer Studies
Thesis code: ORL-80436

WAGENINGEN
UNIVERSITY & RESEARCH

ma
UNIVERSIDAD
DE MÁLAGA

MSc Thesis

Value Iteration for Perishable Inventory Control

Cleo.G.F. Kortenhorst

Studentnumber:
940916469090

MSc Management, Economics and Consumer Studies
Operations Research and Logistics

Supervisors:
Eligius.M.T. Hendrix (WUR)
Karin.G.J. Pauls-Worm (WUR)

Thesis code:
ORL-80436

Preface

Writing a thesis, one of the most terrifying parts of academic education (according to most students). The days are long and the subject is tough. This impression did also apply for me. When I was asked to do my thesis in Málaga, I agreed immediately. And took the plane to Málaga, where Eligius was waiting to show me around. Eligius turned out to be the supervisor, who wanted to help with everything, even though it had nothing to do with study. Together with Inma and Gloria, he made me feel really welcome. After 1 month Maartje arrived, and we made the "terrifying" thesis period, a nice and memorable stay.
Gloria did help me getting started with Python, Eligius was the mastermind who knew a solution for every problem, Karin for long-distance supervising and Maartje turned out to be a funny and crazy room-mate/companion! I want to thank everybody who made my stay in Málaga pleasant and never to forget!

Abstract

This thesis focuses on finding optimal order policies for perishable inventory in supermarkets. By means of value iteration (VI) and simulation, several inventory dynamics are compared. First, several cases were designed: three cases without taking perishability into account, and one case based on perishability. Case 1-3 have the following characteristics, respectively: 1) backlogging is possible, and costs should be minimized; 2) backlogging is not permitted, and profit should be maximized; 3) the customer service level (CSL) should be met, and costs should be minimized. The fourth case, 4) contains products which perish after one day, and the profit should be maximized. The calculation of an optimal order policy is performed by implementing a VI method. The optimal order quantity per inventory level calculated in VI, is used in the simulation. The four cases gave different optimal order-policies. The first three cases showed a lot of similarities, and had the characteristics of an (s, S)-policy. Case 4, containing perishable products, was highly influenced by the fixed order cost k. If the fixed order costs were low, an optimal order policy could be determined. This was not possible for high fixed order costs, the order moments and order quantities turned out to be periodic. For fixed costs which were too high, no orders were placed. The order policy for the low fixed costs did not match the policies in the inventory control chapter. In case 4, we had to deal with the curse of dimensionality. The limited shelf-life causes an increase in state space. To deal with this problem, we had to keep the elements per set of states low.

Contents

Chapter 1

Introduction

1.1 Current Situation

Currently the world population is growing intensively, which means that it gets harder to feed everybody. This problem makes it more important to reduce food waste as much as possible. According to the FAO (2011), one third of world food production goes to waste. Figure 1.1 shows that food waste is the highest when the food is still in the supply chain and food waste is lower at the customer. Especially in Europe, North-America and Oceania the level of waste is high and needs to be decreased (as well at the production, as at the customer). A German research estimates the foodwaste at retailers around 3% of the whole world's waste, when extrapolating (Eriksson, 2012). For the whole European Union, food waste at retailers is estimated around $4,433,000$ tons per year (EC, 2010). About $\sim$ 10% of food wasted at retailers is perishable (fruits and vegetables). These 10% could be divided into two categories; 78% is pre-store waste whilst 22% is in-store waste (Eriksson, 2012).

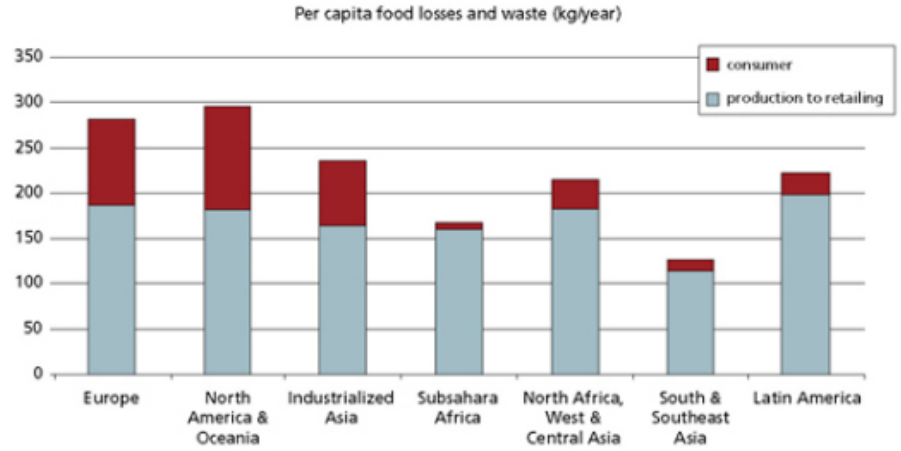


Figure 1.1: Per capita food loss and waste, at consumption and pre-consumption stages, in different regions (FAO, 2011).

Food waste can be caused by several factors depending on conditions and local situations. It is also caused by choices and patterns made during crop production, infrastructure/capacity, marketing, distribution channels, customer purchasing and use of food (FAO, 2011).

Table 1.1 shows waste percentages per perishable food category. What is shown, is that fresh fruit and vegetables have by far the highest waste percentage. This category is of interest in inventory management due to its high waste percentage.

This thesis will mainly focus on the retail-side, trying to figure out how to optimize inventory management for perishable foods to reduce waste. Different types of inventory management are described in the following section.

Table 1.1: Waste percentages for different food product categories at retailer (Eriksson, 2012).

Product categories	[ton]	[%]
Fresh fruits and Vegetables	8,574	5.5
Cheese	1,057	0.528
Dairy	10,931	0.328
Deli	1,243	1.35
Meat	1,380	1.22
Annual total	*23,184*	*2.4*

1.2 Background

This research is continuing on the working-paper of Hendrix et al. (2018), which focuses on finding the optimal order policy for highly perishable foods. During their research they found out that when perishable products have a long shelf-life, the products tend

to act like non-perishable products and the order policy can be adapted to that. The shorter the shelf-life becomes, the harder it gets to find an optimal order policy. The big challenge remains in the complexity of the algorithms, when other perishable products are used and the shelf-life needed in the algorithm changes. However, the algorithm should still be calculable (Hendrix et al., 2018).

1.2.1 Types of food products

Among food products, two types can be distinguished: perishable and non-perishable products. Apart from the use-by or best-before date, there are multiple differences between perishable and non-perishable products, as shown in Table 1.2. In this thesis, perishable products is the group of highly perishable products, e.g. salads, mussels, poultry, fish etc. It is hard to optimize order policies due to uncertain demand in combination with highly perishable products. These products are perishable within 3-5 days. Non-perishable products are in this case seen as products with a shelf-life longer than 3-5 days.

Table 1.2: Definitions of perishable and non-perishable (food-) products, according to Parfitt et al. (2010).

non-perishable food crops	perishable food crops
harvest mainly seasonal, need for long-term storage	possibility of permanent or semi-permanent production, short-term storage needs
preliminary treatment (except threshing) of the crop before storage exceptional	processing of dried products—an alternative to the shortage of fresh products
products with low level of moisture content (10–15% or less)	products with high level of moisture in general between 50 and 80%
small 'fruits' of less than 1 g	voluminous and heavy fruits from 5 g to 5 kg or more
respiratory activity of stored product very low, heat limited	high or even very high respiratory activity of stored products inducing heat emission in particular in tropical climates
hard tissues, good protection against injuries	soft tissues, highly vulnerable
good natural disposition for storage even for several years	products easily perishable, natural disposition for storage between some weeks and several months
losses during storage mainly from exogenous factors (moisture, insects or rodents)	losses owing partly to endogenous (respiration, transpiration, germination) and to exogenous factors (rot, insects)

1.2.2 Order policies

There are multiple order policies, which are practical and efficient for certain inventory situations.

Continuous reviewing and periodic reviewing are two types of monitoring which exist among order policies. During a continuous review, the inventory level is constantly monitored. After reaching the Re-Order Point (ROP or s), a new order is placed. This means that a continuous review policy ((s,Q)-policy and (s,S)-policy) is not time-dependent.

The second review is a periodic review policy ((R,S)-policy and (R,s,S)-policy), which means that the inventory level is measured every pre-determined period R. In both cases, continuous review and periodic review, it is possible to order an amount to refill the inventory up to a certain inventory level (called an Order-up-to-level S), or to order a fixed quantity Q. In this thesis, the inventory level of the supermarket will be measured every day. This means that a periodic review policy is implemented. Both types of reviewing can be encountered in the literature, chapter of Pauls-Worm (2007).
Another challenge of finding the correct order policy is perishability of the products.

Value Iteration and Simulation

Many situations in inventory management are too complex to be able to calculate the optimum analytically. Trial and error is too expensive to find the optimal order policy in real life. That is why value iteration is applied.
Value iteration is used to derive the optimal policy in a **stationary** situation. The outcome of value iteration shows the optimal amount to order, although it takes calculation time. Value iteration is based on Dynamic Programming (DP), with the most important mathematical formulation called the Bellman equation. The Bellman equation (1.1), for a deterministic approach defines the value function V(s) (when the outcome is deterministic/predictable)(Bellman, 1957). A VI contains several components, used to find the optimal valuation. State s represents the state in which the order/production is at that moment. In figure 1.2 this is the inventory level I.
The contribution of changing state depends on the case, whether back-ordering is (not) allowed, or profit/costs need to be focussed on. The used cost function $C(s, x)$ is case dependent and defined for each case in Chapter 3.
$T(s, x)$ is the so-called transformation function, indicating the reached state when performing action x in state s. At each state, decisions x should be made for the next moment. In this case, decision x is the order quantity at state s. The transformation function describes the motion obtained by the whole calculation (*"equation of motion"*).
The Bellman equation is

$$\forall s \in S : V(s) + \pi = \min_{x \in X(s)} \{C(s, x) + V(T(s, x))\}, \tag{1.1}$$

where π is the average profit or cost over a period and V is the so-called valuation function which provides a valuation for state s. $X(s)$ is the action space the decision is taken from. When a Bellman equation is extended to a stochastic situation, the expectation operator is added (Bellman, 1957):

$$\forall s \in S : V(s) + \pi = \min_{x \in X(s)} \{C(s, x) + E(V(T(s, x, e)))\}. \tag{1.2}$$

If value function V is known, it is possible to calculate the optimal policy for a model.

$$\forall s \in S : x(s) = \arg \min_{x \in X(s)} \{C(s, x) + E(V(T(s, x, e)))\}, \tag{1.3}$$

where e is a stochastic event. In our case it is the stochastic demand.
In many cases of retail, a so-called α- service level requirement applies. An α- service level of 0.95 (e.g.) means that in 95% of the periods the demand should be fulfilled (Pauls-Worm, 2007). The reason that the α - service level is used and not a fill-rate e.g. is that for a retailer, usually the final demand is unknown. Monitoring the shelves gives clear information about the inventory level: within a certain α-level it is allowed to go out-of-stock in $1 - \alpha\%$ of the periods and still meet the target service level of $\alpha\%$.

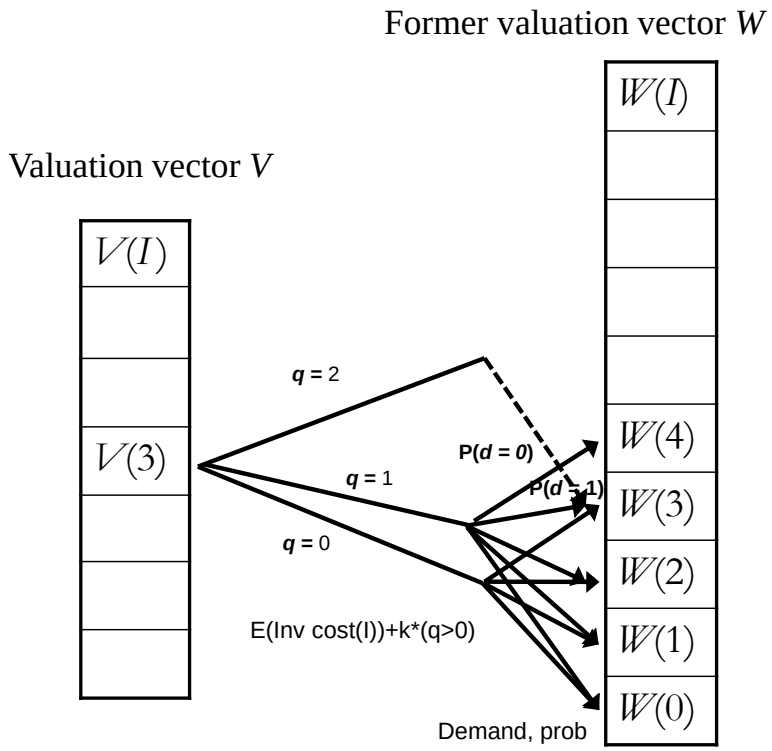


Figure 1.2: From a given state i, given the order quantity and the random demand a new state j will be reached with probability P_d. Based on the workingpaper of Hendrix et al. (2019).

Curse of Dimensionality

According to Bellman (1966), the curse of dimensionality is the exponential increase in volume combined with adding dimensions to the Euclidian space, and causes calculation problems. When the dimension of the model grows, several problems start to exist: The outcomes are not possible to visualize/draw a graph of it. Computing time and general performances of the program (as cited above)(Verleysen and François, 2005). According to Verleysen and François (2005), if every dimension has a certain amount of data, it grows exponentially when adding a dimension (e.g. $1D = 10$, $3D$ will need an amount

of data of 1000 units).

1.2.3 Demand Patterns

Companies which sell slow moving products are more likely to have customers who are willing to wait for backlogging of out-of-stock products. Customers of a supermarket are not willing to wait that long, resulting in lost sales. Phrased in another way, the possibility to backlog or to have lost sales is partially caused by different product categories (Pauls-Worm, 2007). Figure 1.3 shows four theoretically defined demand patterns:

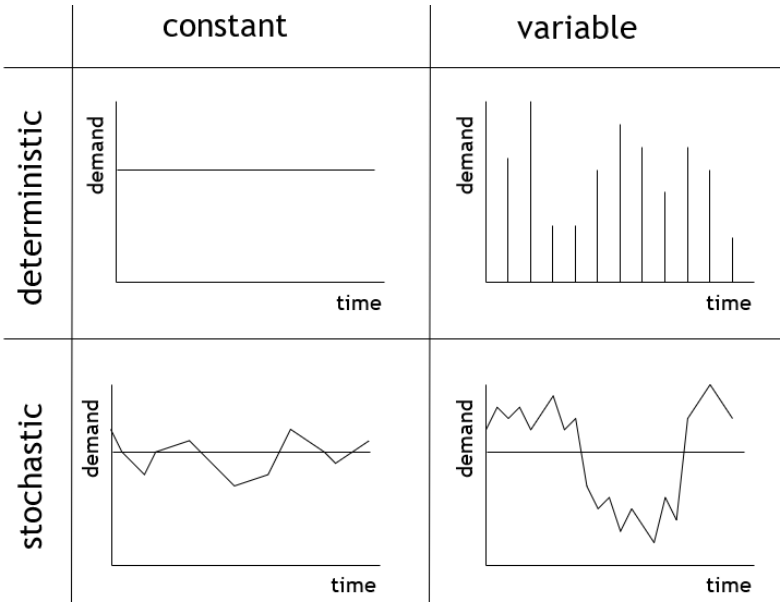


Figure 1.3: Varying types of demand patterns. Among them the stationary stochastic demand studied in this thesis (Pauls-Worm, 2007).

Demand is uncertain in supermarkets. Although one can make a prediction, the demand realisation during a week is uncertain. This means that the demand in a supermarket is stochastic, whilst for deterministic demand it is possible to be able to predict the demand precisely. Within stochastic demand, two demand patterns can be distinguished: stationary demand and non-stationary demand. Stationary (constant) means that the expected demand is constant. Non-stationary (variable) means that the demand is predictable for specified moments over time, but the prediction is not constant (Pauls-Worm, 2007).

1.2.4 Previous Study on Perishability

The problem of perishable product inventory management is known as difficult and also contains many challenges in supermarkets (Van Donselaar et al., 2006). According to

Van Donselaar et al. (2006), the following criteria are used to distinguish between perishable and non-perishable products:
• Perishable products deteriorate quickly at ambient storage conditions and need specific storage conditions to slow deterioration. A freezer is kept out of scope, because of its ability to lower the deterioration until the product reaches the level of non-perishability.
• The fast deterioration rate of perishable products makes it impractical to re-order products with the same date.

Van Donselaar et al. (2006) define perishable products as products with a shelf-life of 30 days or less. Estimated is that roughly 15% of the assortment of a supermarket is perishable, while 85% is non-perishable. Table 1.3 illustrates the criteria mentioned above, as well as other measurable differences between perishable and non-perishable products.

Table 1.3: Van Donselaar et al. (2006) performed a statistical χ - test, to study the differences between perishable and non-perishable products.

	Perishables	Non-perishables	Test statistic
Average weekly sales per product (in CU)	11.9	7.9	116.24
Coefficient of variation of weekly sales	0.345	0.377	27.17
Potential delivery frequency (in days)	1.2	1.3	82.35
Shelf life (in days)	21	240	1050.00
Average case pack size (in CU)	6	10	523.50
Minimum inventory norm (in CU)	3.3	4	47.49

Nahmias (2011) performed a broad study on order policies and perishability. Some of his conclusions were:
• Many articles only focus on the inventory life cycle, but do not take the lead time into account. The lead time could be set to zero. This means that the re-order point can be set to zero and the order quantity can be equal to the so-called Economic Order Quantity. However, it is difficult when the lead time is positive. This causes an infinite dimensional vector of all previous orders placed and their ages (Nahmias, 2011).
• It is well known that for many non-perishable continuous review systems, the optimal order policy is an (s, Q)-policy. However, in case of perishability, when it is possible that inventory drops to zero due to perishability, it is important to adjust the model. The amount Q is ordered after inventory hits zero or drops below s. An order should be made no matter which of the two cases occurs first. In this case, an (s, Q) -policy makes no sense, but a (T, s, S)-policy does. This policy means to order when inventory drops below s, or after time interval T (Nahmias, 2011).

1.3 Research Questions

Currently, it is difficult for retailers to find the right order policy for perishable products. This difficulty in finding the right order policy causes a high waste of food products, which is repeated every order cycle. It is important to find for each inventory situation an appropriate order policy, and to be able to implement this policy. To understand the outcomes of the different situations, it is important to know which order policies exist for perishable products. In this thesis, several cases are designed for different situations. After the designing-phase, the cases are optimized by Value Iteration and checked with simulation. Applying VI, the curse of dimensionality could cause trouble due to large state spaces. These state spaces are caused by having a limited shelf-life. Finding the right policy for a certain situation for (non-) perishable goods using Value Iteration and simulation is the objective of this thesis. The research questions are divided into one main question and two sub-questions to support the main question.

1.3.1 Main Research Question

How can computational Value Iteration (VI) be implemented to optimize perishable inventory control?

1.3.2 Sub-Questions

1. Which kind of order policies exist for inventory control of perishable goods?
2.What is the optimal order policy for non-perishable products, when applying VI?
3.What is the optimal order policy for perishable products, when applying VI?
4.How does the simulation compare to the VI, when the outcomes of the found optimal order policy are taken into account?
5.How to deal with the curse of dimensionality and set-up correct limits when implementing VI?

1.4 Objectives and aims

1.4.1 General aim

The main aim of this research is to find optimal order policies for perishable inventory in supermarkets, to reduce food waste and increase profit (and to continue on the paper of Hendrix et al. (2018)). By splitting up the research question into smaller cases, it is more likely to find a solution and to deal with the curse of dimensionality.

1.4.2 Specific aims

The research is divided into two different approaches and will also be tackled as two separate approaches: A) Theoretically supported research and B) Designing inventory

policies for several small cases.

A) Theoretically supported research:
Firstly, it is important to obtain information about different order policies. Subsequently, literature is consulted about when to use which policy, in case of perishable or non-perishable products.

B) Finding correct inventory policy per shelf-life:
To be able to investigate the main research question, several models are designed, for different inventory situations. Also perishability is taken into account, which makes it possible to find the correct policy for certain situations (e.g. different shelf-life, back-ordering, customer service level, or making profit). These findings should match the obtained theory. Therefore in this approach it is important to match the theoretical order policies with the models (value iteration) made.

1.5 Research Design and Methods

1.5.1 Methods

The sub-research-questions are studied using two approaches: literature-based questions, and programming-based questions. *Question 1* is investigated using a literature study. *Question 2* and *3* are investigated by using an approach of coding/programming and is completely implemented in Python. Before developing a value iteration for several models, it was important to know how to use Python. This is done by following an e-course and by practising small codes. After discovering Python, it was time to perform the VI and the simulation models. *Question 2* does not take shelf-life into account, because of its non-perishability (case 1-3), while *Question 3* does take shelf-life into account (case 4). First the VI is ran, to find the optimal order policy. After finding the optimal order policy, it is used as input for the simulation (*Question 4*). This simulation should match the VI to be sure the policy is optimal. This approach is used for all cases, perishable or non-perishable. The newly written model is explained in an algorithm. The results from this literature study are used to compare the findings obtained from the programming model. This comparison makes it possible to know whether the designed model makes sense or not.

The model results in an outcome containing multiple dimensions, as explained in the section above. These dimensions are caused by an increase of state-space and this increase makes it harder to get a feasible/calculable program. Which results in *Question 5*: How to deal with the dimensionality? This challenge is faced by finding techniques in literature, to reduce the state space and by that the program's running time. After investigating the sub-questions (aka the approaches), it is possible to summarize the findings with respect to the main research question.

Chapter 2

Inventory Control

Inventory is kept because of the fact that most of the times, it is not possible to have every product arriving at the exact time the product is needed. Inventory is also kept to deal with uncertainties in demand and supply, and to deal with seasons etcetera. But on the other hand, inventory could also cause losses, due to e.g. expiring inventory. Inventory ties up money or working capital which could not be invested otherwise, to make profit (Pauls-Worm, 2007). In other words, it is important to have a good equilibrium between holding costs and fixed order costs. Inventory could be perishable or non-perishable, which requires different optimal order policies.

Products with a short shelf-life are highly vulnerable to become waste. To reduce the amount of waste, the following options can be used (Van Donselaar et al., 2006):
- Reduction of leadtime and/or review period;
- Demand substitution;
- Limited assortments.

To realize waste reduction, an optimal order policy is important. Especially to be able to meet demand and reduce waste. In this thesis, the order policies for both perishable and non-perishable goods are examined.

2.1 Continuous Review Policies

A continuous review policy means there is a continuous review of the inventory level ($R = 0$). All the following order policies within this paragraph are based on ordering when the inventory drops below a certain level. This level does not depend on periodic orders (Pauls-Worm, 2007). The continuous review policy of Tijms and Groenevelt (1984) reviews the inventory position after each demand transaction.

2.1.1 (s, Q)- order policy

As shown in Figure 2.1 an (s, Q)-order policy graph has the characteristic that both the s and the Q are set and do not variate. The re-order point (s or ROP) could be determined by

$$ROP = \mu t_l, \tag{2.1}$$

of which μ is the average demand and t_l the lead-time. This formula is correct when the demand is known and constant. If not, safety stock is added or there are possibilities to back-order. Hill and Johansen (2006) wrote the following: In a lost sales model for non-perishables with a positive product lead-time it is shown, under a common linear costs structure that the optimal policy is restricted by a maximum order quantity Q (an (s, Q)-order policy), this is confirmed by Haijema and Minner (2016). In principle, an (s, Q)-order policy follows the continuous review method (Pauls-Worm, 2007). According to Fawcett et al. (2014), the (s, Q)-policy diagram should follow its characteristic sawtooth diagram, as shown in figure 2.1. The (s, Q) diagram has every cycle the same value Q (Fawcett et al., 2014). Expressed in mathematical terms (Haijema and Minner, 2016), the order quantity q is given by

$$\mathrm{q} = \begin{cases} \text{if I} < \text{s} & \mathrm{Q} \\ \text{otherwise} & 0 \end{cases}. \tag{2.2}$$

Besides of the shape of the graph as characteristic, it also exists of two "bins". After the first bin is empty the second one is opened/used and the first one requires re-ordering. One drawback of an (s, Q)-system is that it is possible that the demand causes problems within the inventory level. The inventory level could show the following pattern: $ROP - Q < 0$. This results in the problem that after re-ordering the order quantity Q, the ROP is still not met. This means that back-ordering is required even immediately after the order quantity Q has entered the shop, or the demand turns out as lost sales, when there is no back-ordering possible. As result, the situation remains out-of-stock (Pauls-Worm, 2007).

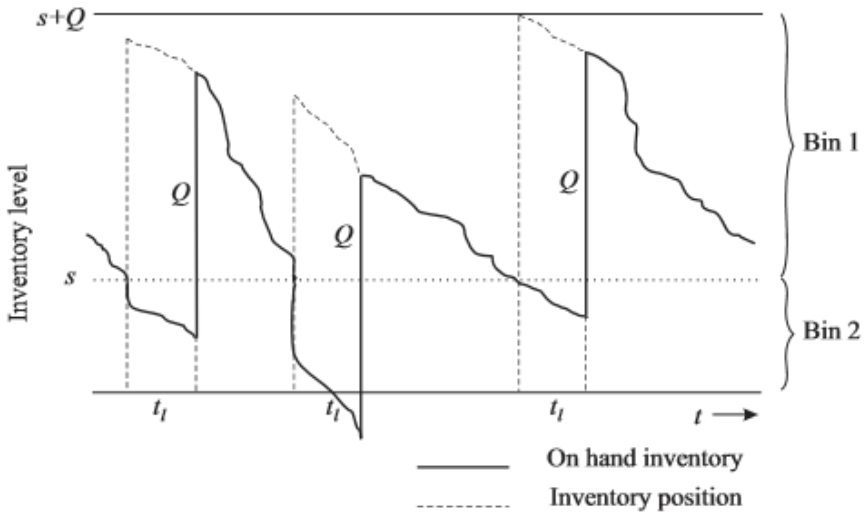


Figure 2.1: A typical example of an (s, Q)-graph, with the sawtooth shape and the two bins (Pauls-Worm, 2007).

2.1.2 (s, S)- order policy

An (s, S)-policy is used when the company wants its inventory level maintained, which means that after reaching the re-order point s the inventory level is brought back to a predefined maximum inventory level S. This means that the order quantity Q is adapted to that, every time an order is placed (Pauls-Worm, 2007). Expressed in mathematical terms (Haijema and Minner, 2016), the order quantity q is given by

$$\mathrm{q} = \begin{cases} \text{if I} < \text{s} & \text{(S-I)} \\ \text{otherwise} & 0 \end{cases} . \tag{2.3}$$

Figure 2.2 shows a graph where an Order-up-to level policy is depicted. This order policy is widely used (Hendrix et al., 2018). To have an optimized order-up-to level policy for retailers, it is worth knowing the inventory costs and the lead time of the product are. When the fixed costs decrease, the number of orders placed by the retailer will be higher and the orders will be more precise (Bertazzi et al., 2002). As soon as the inventory level reaches s, an order will be placed. This order will be delivered after the lead time (t_l) has expired and the inventory level will be replenished until S level minus the products sold during t_l. The orders are more precise than a (s, Q) order policy. The downside of an (s, S) system is the computational effort it takes, as well as the possible errors that could be made during the ordering of variable orders (Pauls-Worm, 2007). According to Tekin et al. (2001) is the (s, S)- order policy optimal for non-perishables facing Markovian demand.

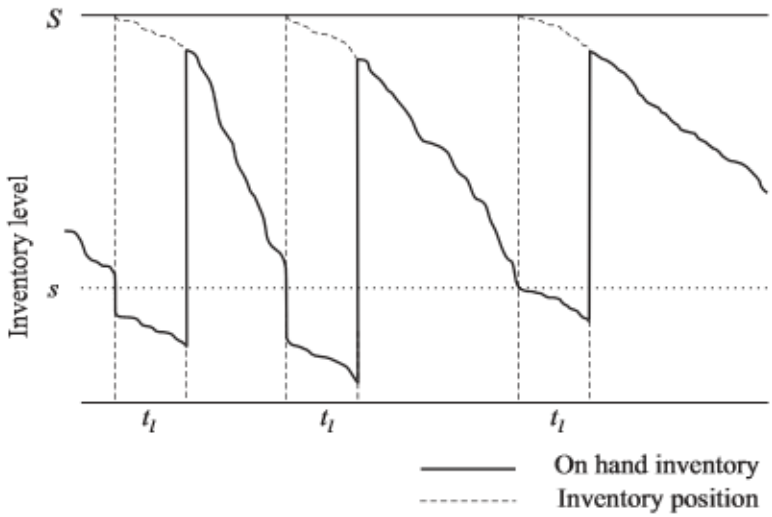


Figure 2.2: Order-up-to level policy, where after re-ordering (at s), the inventory level is refilled to S (before lead time) (Pauls-Worm, 2007).

2.1.3 Continuous review order policies for perishable products

(S, q_{min})- and (S, Q_{max})- order policies

These two policies are, up to a certain level, equal to (s, Q)- order policy. The (S, q_{min})-order policy, needs a minimum amount of q ordered per batch, up to level S. Expressed into a mathematical statement (Haijema, 2013): The order quantity is expressed in mathematical terms by

$$q = \max(q_{min}, S - I). \tag{2.4}$$

While the $(S, Q_{max}$)-order policy, on the other hand, has a maximal amount to be ordered per batch. Limiting q reduces the risk of perished inventory.

$$q = \min(Q_{max}, S - I), \tag{2.5}$$

determines the order quantity q according to (Haijema, 2013). Figure 2.3 shows the two policies which consist of two parameters (Haijema and Minner, 2016).

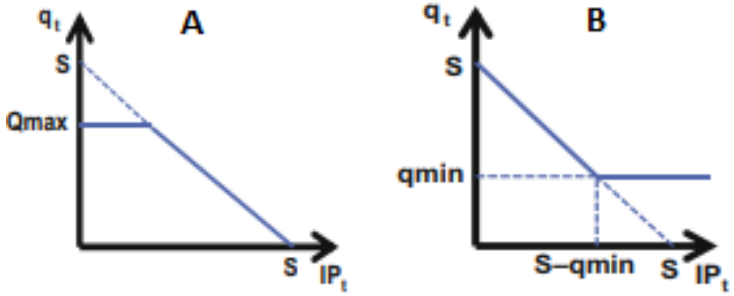


Figure 2.3: A is the (S, Q_{max})- order policy (IP_t is Inventory), with inventory on the x-axis and order quantity on the y-axis. B is the (s, q_{min})- order policy with the same division of axes as A (Haijema and Minner, 2016).

(s, S, q, Q)-order policy

The (s, S, q, Q)-policy has been developed by Haijema (2013), and is a combination between an (s, S)-policy and an (s, Q)-policy. The (s, S)-policy is inserted as policy and the q and Q are used as minimum and maximum order quantity (Haijema, 2013). Haijema (2013) tested the (s, S, q, Q)-policy against an (s, S)-policy and a pre-defined optimal cost policy. It appeared that the (s, S)-policy was 8% above optimal costs, while the (s, S, q, Q)-policy was only 3.8% above optimal costs. They concluded that the bounds in order quantity smoothed the age distribution, which reduced out dating. They also concluded that changing the shortage costs had less influence than changing fixed costs. Changing fixed costs influenced the amount of orders per day. The (s, S, q, Q)- order policy is a big improvement in cases where order frequency is low and out-dating is high (Haijema, 2013).

(Q, r, T)-order policy

The (Q, r, T)-order policy is based on a continuous review of perishable inventory. The (Q, r, T)-order policy is developed by Tekin et al. (2001). There are two different triggers in this order policy to order new products: When the inventory level goes below a treshold value r, or when a time span T has expired. After an order is placed, a new time period starts. The order quantity Q is always of the same size.
The expiring time span T has to do with the shelf-life of perishable products. When a product is expired, new products are necessary and an order is required. The contribution of this order policy is, the re-ordering is not only stock level dependent but also the remaining shelf-life of the products in stock are taken into account (Tekin et al., 2001).

2.2 Periodic Review Policies

A periodic review policy depends on time, contrary to a continuous review policy. In this system every R units of time the inventory levels are checked (Pauls-Worm, 2007). Tijms and Groenevelt (1984) state that in a periodic review policy, the inventory position is only checked at the beginning of the R-th period. According to Prak et al. (2015), periodic review policies are more realistic than continuous review policies, because even with current automatic counting systems continuous review policies are not feasible.

Chopra and Meindl (2016) gave an expression for the average lot size of a periodic review policy

$$Q = D_T, \tag{2.6}$$

where D_T is the demand per time period T.

2.2.1 (R, S)- order policy

This policy has two important parameters, R, which is a timebound period and S which is the order-up-to level. It is important that S should cover the demand during review period R as well as during leadtime t_l. The inventory level of the (R, S)-policy tends to be higher than the inventory levels of continuous review policies. However, due to the fact that retailers could only order at fixed times, it is a more practical policy (Pauls-Worm, 2007). Figure 2.4 visualizes what happens with the inventory level when a (R, S)-policy is performed. This policy also exists for fixed order quantity Q instead of an order order-up-to level S (an (R, Q)-order policy).

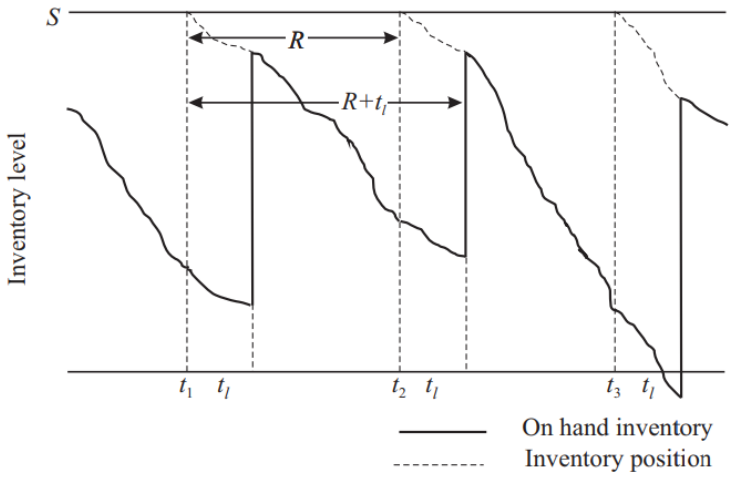


Figure 2.4: An (R, S)-policy, in which R is timebound and S is the order-up-to level, t_l is the lead time (Pauls-Worm, 2007).

2.2.2 (R, s, S)- order policy

The (R, s, S)-policy combines two, before mentioned, (s, S)- and (R, S)-policies. If the inventory level is below re-order point s and period R is expired, an order is placed. When after a period of R, the inventory level is still above s, no order is given. Figure 2.5 visualizes what happens with the inventory level when a (R, s, S)-policy is performed. The (R, s, S)-policy is the system with the lowest costs, when the optimals are found. However, the computational efforts necessary to calculate these optimals is high (Pauls-Worm, 2007).

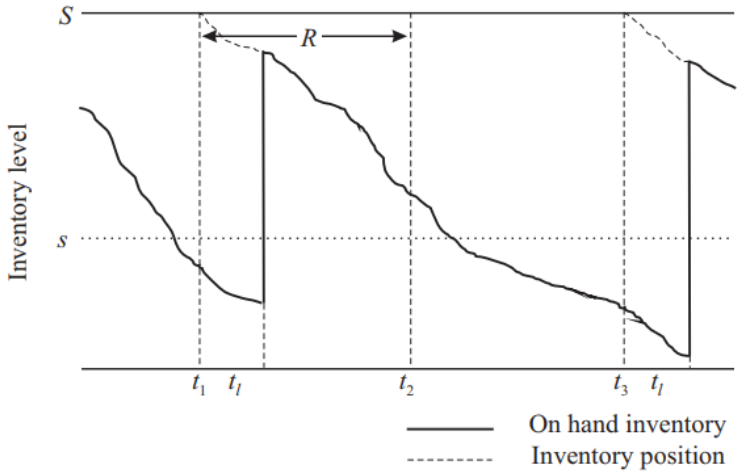


Figure 2.5: An (R,s,S)-policy, in which R is time bound, reorder level s and S is the order-up-to level, t_l is the lead time (Pauls-Worm, 2007).

2.2.3 Periodic review order policies for perishable products

(s_d, S_d, q_d, Q_d)- order policy

In this previous section the continuous review s, S, q, Q- order policy was described. According to Haijema (2013), this order policy is also applicable when inventory is periodically reviewed. This is possible by adding time d indexes, instead of an (s, S, q, Q)-order policy, an (s_d, S_d, q_d, Q_d)- order policy is used (Haijema, 2013).

EWA order policy and (R, s, nQ)-order policy

Broekmeulen and Van Donselaar (2009) compared two types of order policies: the EWA policy and the base (R, s, nQ)-order policy. In their research the products are perishable, the safety stock is constant, the review system is periodic, back-ordering is not allowed, and they use batches. These requirements are used for both policies. The (R, s, nQ)-base order policy, where s_t (set every day) is the reorder level which is checked every time period R. If the inventory level is below the reorder level, a number of case packs n_t is ordered, with size Q. The difference between EWA policy and the (R, s, nQ)-order policy is the estimated amount of out-dating, which is incurred in the EWA order policy. When assuming First In First Out $FIFO$, the EWA order policy is best for short remaining shelf-life, long lead time, long review period, high out-dating costs and high lost sales cost ratio(Broekmeulen and Van Donselaar, 2009). The (R, s, nQ)- order policy has similar characteristics as the Q, r, T-order policy. However, the (R, s, nQ)-order policy does only measure the inventory level at the end of each time period. While the (Q, r, T)- order policy measures continuously (the inventory level), but also keeps

the age of the inventory in mind by periodically disposing old products.

2.3 Conclusion

Table 2.1 is a summary of the order policies described above. The table gives the most important characteristics per order policy, for both perishable and non-perishable products. The non-perishable product order policies are based solely on the inventory level, or a time period. Whereas the perishable product order policies focus on the inventory level and the shelf-life. Figure 2.6 classifies the order policies are based on a periodic review of inventory, or a continuous review of inventory. And if the order quantity Q is fixed or variable (order-up-to level S). In this thesis it is interesting to compare the order policies with the outcomes of the models. How do the non-perishable product cases compare to the non-perishable product order policies described in this chapter? And how do the perishable products compare?

Table 2.1: All explained order policies, with their deviating characteristic(s). Focussed non-perishable N and perishable P products.

Continuous review policy	
(s, Q)-order policy	N. Easy to apply.
(s, S)-order policy	N. High computational effort.
(s, S, q, Q)-order policy	P. Reduces out-dating.
(Q, r, T)-order policy	P. Two triggers: inventory level and shelf-life.
Periodic review policy	
(R, S)-order policy	N. Practical approach.
(R, s, S)-order policy	N. High computational effort.
(s_d, S_d, q_d, Q_d)-order policy	P. Reduces out-dating.
EWA -order policy	P. Short shelf-life, long lead-time.

Order frequency	Fixed (Q)	Order quantity: Variable (S)
Continuous	*(s,Q)* *(s,S,q,Q)* *(Q,r,T)*	*(s,S)*
Periodic	*(R,s,nQ) or EWA* *(R,Q)* $s_t S_t q_t Q_t$	*(R,s,S)* *(R,S)*

Figure 2.6: All explained order policies separated into periodic or continuous reviewing and fixed order quantity or variable order quantity.

Chapter 3

Models: Value Iteration and Simulation

To be able to investigate the research questions about finding the optimal order policies, for as well non-perishable products as perishable products, small cases are designed. First, three cases are designed are made for non-perishable products. Secondly, a design is made for perishable products. For all cases an order policy a Value Iteration method is developed and to evaluate the order policy a simulation is constructed.
Figure 3.1 shows the order of events in this research. The dynamics of the inventory I_b and I_{end} is given by (3.1), (3.2) and (3.3). During the day, customers buy products (demand). Demand ($d \sim poisson(\mu)$) is withdrawn from the start inventory I_b. After d is withdrawn from I_b, end inventory I_{end} is reached. For demand of the next days, an order is placed (one day in advance). This order quantity q is added in the time between closing and opening the store. The inventory for the next day follows equation (3.3). If back-ordering is permitted, the start inventory I_b is not necessarily positive. The order-quantity is based on the calculated optimal order quantity Qop, which is based on the inventory level of the day before.

$$\mathbf{I}_{end,t} = \mathbf{I}_{b,t} - \mathbf{d} \tag{3.1}$$

$$q = Q(I_{b,t-1}) \tag{3.2}$$

$$\mathrm{I}_{b,t} = \begin{cases} \text{back-order allowed} & I_{end,t-1} + q_{t-1} \\ \text{back-order not allowed} & (I_{end,t-1} + q_{t-1})^+ \end{cases} \tag{3.3}$$

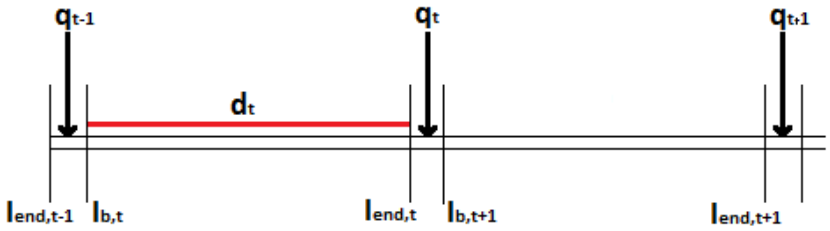


Figure 3.1: Event sequence in which order and demand occur. I_b represents the inventory at the beginning of the day, just before the store opens. This is when inventory is measured and a new quantity q is ordered. During the day, customers buy products (demand, d) from inventory. When the shop closes its doors, I_{end} is realised. Between closing and opening the doors, new inventory arrives at the shop. This new inventory is the order quantity (q) of the day before.

3.1 Non-perishable Products: Case 1

This case models a situation in which a retailer is allowed to perform backlogging. Backlogging is used when the retailer turns out-of-stock and the customers are willing to wait for re-ordered products. For the retailer it means re-order costs, but backlogging prevents loss of sales. Depending on the willingness of the customer to wait for his/her products, it is expected that the inventory level could be held around the average demand level μ. One possible reason to keep the inventory level a bit above the average demand level is the fact that back-ordering is more expensive than holding inventory. This means that the model should find an equilibrium between holding costs h, fixed order costs k and back-order costs c. Table 3.1 shows the values corresponding to the used input data (costs and average demand level).

Table 3.1: Used input data.

c	1
μ	2
h	0.25
k	4

Value Iteration of Non-Perishable Products: Back-ordering allowed
The model is based on a VI which determines the optimal cost function, which is a procedure to find the optimal order policy.A value iteration contains a contribution,

expectation, previous costs and current costs. For this case, the contribution,

$$C(I_b, q) = k(q > 0) + E(h \times (\mathbf{I}_{end})^+ + c \times (-\mathbf{I}_{end})^+), \tag{3.4}$$

gives the costs in state s taking action q. The transition function for this case which describes the transition from one state to the other is,

$$T(I_b, q, d) = I_b + q - d. \tag{3.5}$$

The transition is used in each valuation of an action q,

$$v(q) = \{C(I_b, q) + \sum_{d=0}^{\infty} P_d W(T(I_b, q, d))\}. \tag{3.6}$$

The valuation $V(I_b)$is reached by taking the minimum of $v(q)$,

$$V(I_b) + \pi = \min_q\{v(q)\}, \tag{3.7}$$

Where $W(I_b)$ (3.6) represents the former valuation vector (3.7). $W(I_b)$ is implemented into the calculation for the valuation vector of the following iteration $V(I_b)$. The daily average costs of the policy are given by $\pi = V(I_b) - W(I_b)$. The optimal order quantity of state I_b can be determined by the following formula,

$$Q(I_b) = \arg\min_q\{v(q)\}. \tag{3.8}$$

The value iteration continues until the difference of vector V and vector W is the same for all state values representing the average daily costs π. The variation of the difference over the state value is called the span,

$$Span = \max_I\{V_I - W_I\} - \min_I\{V_I - W_I\}. \tag{3.9}$$

The optimum is based on costs: order costs, inventory costs and backlogging costs. The optimal order policy is determined by (3.8). Algorithm 1 shows the design of the VI of case 1. Algorithm 3 (Appendix) gives a more elaborate explanation of the VI code.

Algorithm 1 is based on Algorithm 3 (Appendix) and explains how to apply a VI for non-perishable inventory for which backlogging is permitted.

1: Use μ, k, h, c
2: $W(I_b) = k$ and $V(I_b) = 0$
3: **while** Span (3.9) $> \epsilon$ **do**
4: **for** all inventory levels I **do**
5: **for** all order quantities $q = 0 \dots \overline{Q}$ **do**
6: Determine $C(I_b, q)$ (3.4)
7: $v(q) = C(I_b, q)$
8: **for** all demand $d = 0 \dots D$ **do**
9: Determine $T(I_b, q, d)$ (3.5)
10: Compute $v(q) = v(q) + P_d W(T(I_b, q, d))$
11: $V(I_b) = \min_q\{v(q)\}$
12: $Q(I_b) = arg\min_q\{v(q)\}$
13: Average **Costs**: $\pi = V_i - W_i$

After the VI (for each case), the results are projected in two graphs. One shows the optimal order policy as function of the inventory level I. The other graph shows the probability of occurrence per inventory level. This probability of occurrence is based on a Markov matrix. This theory is described by Bellman (1957).

Simulation of Non-Perishable Products: Back-ordering allowed
For the simulation the optimal order policy $Q(I_b)$ is used. This order policy is obtained during VI of case 1. A comparison of the costs found in the VI and the simulation model is made. These costs should turn out equal. After these costs are found equal, the simulation gives the optimal outcome according to the VI. During the simulation the frequency of the inventory level is measured and could be compared to the probability of occurrence found during VI. Algorithm 4 (Appendix) gives a elaborate explanation of the simulation.

3.2 Non-perishable Products: Case 2

Case 2 has quite a different set-up than case 1: backlogging is not possible (due to unwillingness of customer) and profit is optimized. Again, two ways of calculation are used: VI and a simulation with the optimized order policy. For case 1 it was expected that μ would be (almost) equal to the inventory level, due to the possibility of backlogging. In case 2 it is not possible to backlog. If the shop runs out of stock, the customer buys his/her products at another store. To keep the costs as low as possible, the VI model has to find an equilibrium between lost sales and inventory costs. It is expected that the inventory will be higher than the expected demand μ. Table 3.2 shows the used input data.

Table 3.2: Used input data.

μ	2
h	0,25
k	4
f	3

Value Iteration of Non-Perishable Products: Optimization of Profit
Case 2 maximizes profit taking a sales component into account. The contribution is given by,

$$C(I_b, q) = E(f \times \min(d, I_b) - h(\mathbf{I}_{end})^+ - k(q > 0)). \tag{3.10}$$

The transition from one state to another is based on inventory, order quantity and demand. However, in this case it is not possible to have a negative transition. This is due to the fact that back-ordering is not possible. The state transition is described by

$$T(I_b, q, d) = (I_b + q - d)^+. \tag{3.11}$$

The valuation calculation of $v(q)$ with order quantity q is given by

$$v(q) = C(I_b, q) + \sum_{d=0}^{\infty} P_d W(T(I_b, q, d)). \tag{3.12}$$

The optimized valuation, is the maximization of $v(q)$,

$$V(I_b) = \max_q \{v(q)\}. \tag{3.13}$$

The optimal order quantity is the index of the optimal value calculated with formula (3.13),

$$Q(I_b) = \arg\max_q \{v(q)\}. \tag{3.14}$$

In the pseudocode of algorithm 1 the use of (3.4-3.8) is replaced by formulas (3.10-3.14). Line 11 and 12, of algorithm 1, need to be changed from minimization to maximization. A more detailed approach can be found in the Appendix (Algorithm 5).

Simulation of Non-Perishable Products, based on Profit
The simulation is based on making profit instead of solely counting the costs. This means profit is taken into account, and back-ordering is not allowed. Furthermore, the simulation is the same as the simulation performed in Case 1. An extended simulation could be found in the Appendix (Algorithm 6).

3.3 Non-perishable Products: Case 3

A big difference what Case 3 distinguishes from the other cases, is that it focuses on the α- service level. The focus in this case is not to optimize profit or only costs, but also to meet a certain percentage of customer demand. This means that in α of the cases the retailer has sufficient inventory to answer demand. Important for both VI and simulation is that the calculations contain more probabilities. Table 3.3 shows the used input data.

Table 3.3: Used input data.

μ	2
h	0,25
k	4
α	0.9

Value Iteration of Non-Perishable Products: α- service level

First the minimum order quantity (s) is defined based on the α-service level. The minimum order quantity s should be large enough to answer the demand. When s is known, the VI is performed. Because s matches the α- service level, the entire VI matches the α-service level. This only applies when minimum order quantity $\underline{q}$ is equal or higher than s_0. If this criterion is taken into account, the VI could be performed as in case 1, by minimizing the costs.

The probability of getting into a certain new inventory state is P_{end}, i.e. the probability to get into state j. Figure 3.2 shows two states of inventory, the current inventory level, state i, and the next inventory level, state j. The first probability is demand (d), when $i = 2$, the fulfilled demand could be $0, 1$ or 2. This means that next state j is in $0, 1$ or 2 before q arrives, meaning the inventory j contains $0, 1$, or 2 products. After q has arrived the whole graph shifts up in inventory level (I_b).

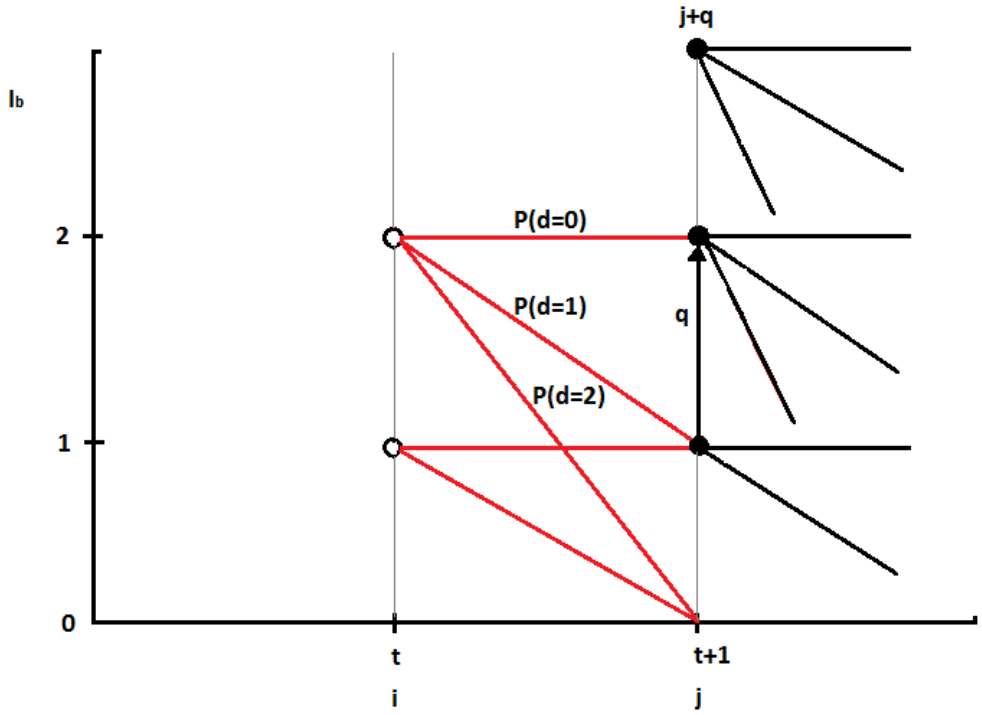


Figure 3.2: The probability in which new state j the inventory level will end up, after if leaves the old state i.

Formula (3.15) reflects one of the goals of this case: α-service level. The probability that the demand is lower than the new inventory and order quantity together, should be larger than α. But this formula only accounts for one certain change of state. Whereas formula (3.16) is based on a double probability calculation and put into a matrix, in which i and j are not predefined.

$$P(d \leq j + q) \geq \alpha \tag{3.15}$$

As stated, the following formula is used to calculate the probability of beginning and ending up in a certain state. The calculation is based on the begin inventory, this means that i is the inventory I_b of the first state, and j is the end inventory I_{end} plus the order quantity. The formula will show a probability matrix,

$$P_{end}(I_b, q) = \sum_{j=0}^{i} P(\psi_{ij} P(d \leq j + q)) \geq \alpha, \tag{3.16}$$

where $\psi_{i,j}$ represents the probability end inventory j, if i is the begin inventory. When the probability CSL is met, in at least α of the times, the minimum order quantity $\underline{q}$ is

set. The rest of the algorithm shows the same characteristics as case 1. The contribution are all the costs made during a change in state:

$$C(I_b, q) = k(q > 0) + E(h \times (\mathbf{I}_{end})^+ + c \times (-\mathbf{I}^+_{end}). \tag{3.17}$$

The transformation shows what happens in a change of state,

$$T(I_b, q, d) = (I_b + q - d)^+. \tag{3.18}$$

The theory explained above is applied in the VI, the pseudo-code is written in Algorithm 2. The more elaborate algorithm can be found in the Appendix (Algorithm 7).

Algorithm 2 A VI for non-perishable inventory with α-service level requirement. Based on Algorithm 7 (Appendix).

Use μ, k, h, c
$W(I_b) = k$ and $V(I_b) = 0$
Calculate P for end inventory (3.16)
such that $CSL \geq \alpha$
Calculate minimum order quantity $\underline{q}(I_b)$
while Span (3.9) $> \epsilon$ **do**
    determine order quantity $\underline{q}$
    for all inventory levels I_b **do**
        for all order quantities $\underline{q} \dots \overline{q}$ **do**
            Determine $C(I_b, q)$ (3.4)
            $v(q) = C(I_b, q)$
            for all demand $d = 0 \dots D$ **do**
                Determine $T(I_b, q, d)$ (3.5)
                Compute $v(q) = v(q) + P_d \times W(T(I_b, q, d))$
        $V(I_b) = \min_q\{v(q)\}$
        $Q(I_b) = arg\min_q\{v(q)\}$
        Determine $Span(V, W)$ and copy W to V
Average **Costs** $= V_i - W_i$

Simulation of Non-Perishable Products: α- service level

One important component in this simulation is the service level calculation (SL-counter). Every simulated day, the SL-counter keeps track on whether demand is met, or not. After the simulation, the service level is determined, to check whether the α-service level is met. The simulation model can be found in the Appendix (Algorithm 8).

3.4 Perishable Products: Case 4

Case 4 is comparable with Case 2, as it aims to optimize profit. The difference is that in this case, the inventory perishes after 2 days. This means that the inventory could

be zero days old (I_0), or one day old (I_1). Table 3.4 shows the used parameter values. Customer demand is expected as stochastic and First-In First-Out (FIFO). After an order is placed, the products from inventory I_1 are used first. When the old inventory I_1 is sold, the products of the fresh inventory I_0 are used.

Table 3.4: Used input data.

c	1,0
μ	2,0
h	0,25
k	4,0
f	3,0

Value Iteration of Perishable Products: Optimization of Profit
Figure 3.3 shows what happens with the valuation V when there is no fresh inventory $I_0 = 0$ (V_{00} or V_{0I_1}). This means the inventory level of W (in the next state of I) is equal to 0, but is refilled with q ($I_0 = q$).
The VI of Case 4 has similarities with the VI of Case 2, however both inventory levels (I_0 and I_1) should be inserted into the calculation of case 4. The contribution in sense of profit is the same as for case 2

$$C(I_0, I_1, q) = E(f \times \min(d, I_0 + I_1)) - h(I_1 - k(q > 0)). \tag{3.19}$$

In the transition state $T(I_0, I_1, q, d)$, two inventory levels are given by,

$$T(I_0, I_1, q, d) \ = \begin{cases} I_1 = & (I_0 - (d - I_1)^+)^+ \\ I_0 = & \text{q} \end{cases} \tag{3.20}$$

Before calculating the optimal valuation, an optimization should happen. $v(q)$, is a valuation calculation for all probable kinds of demand and the matching inventory and order quantity. The valuation $v(q)$ of the order quantity q is given by

$$v(q) = C(I_0, I_1, q) + \sum_{d=0}^{\infty} P_d W(T(I_0, I_1, q, d)). \tag{3.21}$$

After the calculation of $v(q)$, the values are maximized in order to find the optimal solution. This optimization is based on the state of inventory,

$$V(I_0 I_1) = \max_q \{v(q)\}. \tag{3.22}$$

The optimal order quantity is given by

$$Q(I_0 I_1) = \arg\max_q \{v(q)\}. \tag{3.23}$$

The case 4 value iteration requires to consider the inventory levels I_0 and I_1 separately. Throughout the pseudo-code, formulas (3.19-3.23) are used in line 11 and line 12 maximisation takes place rather than minimisation.

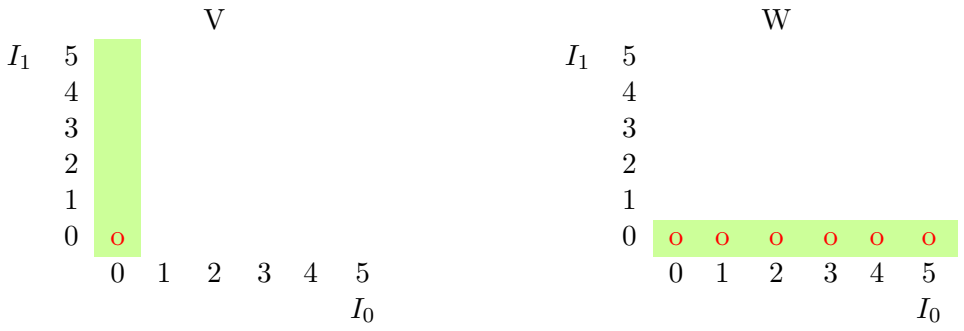


Figure 3.3: Shows the state values to determine V, which represents state $(I_0, I_1 = (0, 0)$, red circle o). The state values $(0, I_1)$ in green transit to state values $(q, 0)$ represented on the right.

For the state of $C_{I_0 I_1}$, it is complicated to know in which state of W the inventory level will end up, this is visualized in Figure 3.4. Important to realize is that demand first has to be subtracted from I_1. After I_1 reaches 0, the inventory of I_0 is used. This means that in the next state, W, the new I_1 is based on the old I_0. While the new $I_0 = q$, which is the same as the other example (shown in Figure 3.3).

A more extended algorithm is available in the Appendix (Algorithm 9).

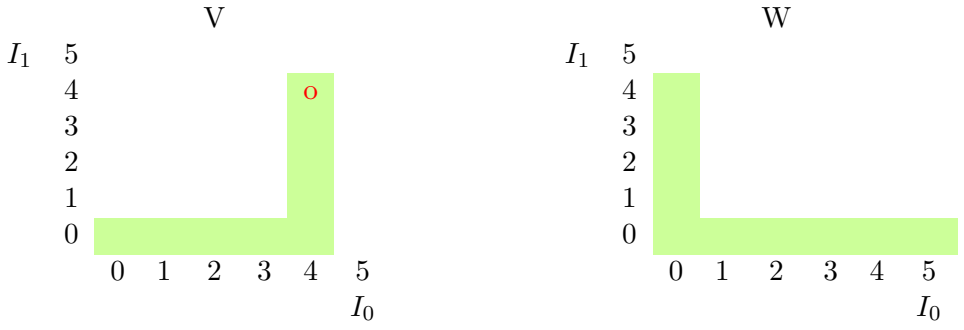


Figure 3.4: The red circle (o) in the V graph presents V_{44}(randomly chosen), the green bar presents where the red circle could go. First, the red circle goes down the I_1-axis ($V_{I_0 0}$). Second, the red circle goes to 0 over the I_0-axis (V_{00}). Depending on whether the red circle reached the situation with the I_0-axis, the next state, in W, is affected. This means that the I_1 of W depends on the inventory level I_0 of V, after demand is subtracted. $I_0 = q$ in state W.

Simulation of Perishable Products: Optimization of Profit

Fresh inventory I_0 is (similarly to the VI) kept as inventort and old unsold inventory I_1 perishes. This also accounts for the optimal Q values found during the VI. The optimal Q values are used in the simulation, to be able to check the output of average profit (e.g.) of the VI. The simulation also counts the frequency of occurrence of the inventory levels I_0 and I_1. The extended algorithm can be found in the Appendix (Algorithm 10).

Chapter 4

Results and Discussion: Value Iteration and Simulation

4.1 Non-perishable Products: Case 1

4.1.1 Results

In this case back-ordering is allowed. Products are not perishable and the input of table 4.1 is used. After running the VI and the simulation, a difference δ of 0.006 is found (table 4.2).

Table 4.1: Used input data.

c	1
μ	2
h	0,25
k	4

Table 4.2: Average costs of order policies, when back-ordering is permitted. For VI and simulation.

Average Costs	
Value Iteration	2.011
Simulation	2.017
δ	0.006

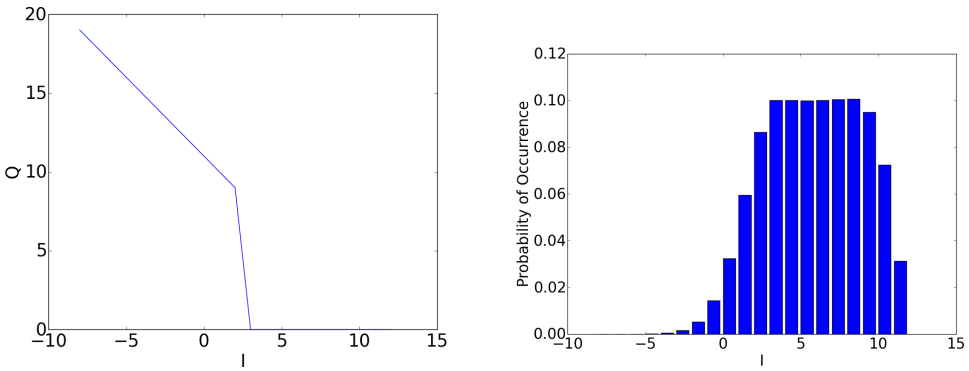
Value Iteration

Within this model a minimum inventory level of -8 is used and a maximum inventory level of 19. The following optimal order quantities Q_{op} are found for those parameter values:

$$Q_{op} = 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0$$

where $Q_{op}(-8) = 19$, as depicted in Figure 4.1a shows Q_{op} against the current inventory level. Below re-order level $s = 2$ an order should be placed. Figure 4.1b shows the probability of occurrence per inventory level.

To know the stationary state inventory (used for the probability of occurrence), an extra step is required. To determine the possible inventory levels, the eigenvector matrix of Markov is used. For this matrix the optimal policy is used, which is gained after performing the VI. The outcome is shown in Figure 4.1b. More information about the eigenvector of a Markov matrix could be found in the Appendix (with a theoretical example).



(a) Optimal order quantity $Q_{op}(I)$ as function of begin inventory I. Notice that I can be negative due to back-ordering.

(b) Shows the probability of occurrence to have a certain amount of inventory in stock, as Markov stationary state.

Figure 4.1: Optimal order quantity and probability of inventory when performing VI.

Figure 4.2 shows what happens when the fixed costs are varied. The value of the fixed order cost is varied in a range from 0 until 10. Every time the fixed order costs increase, the amount of products per order Q increase.
Different values for the fixed order cost k result in the order policies as sketched in Figure 4.1a.

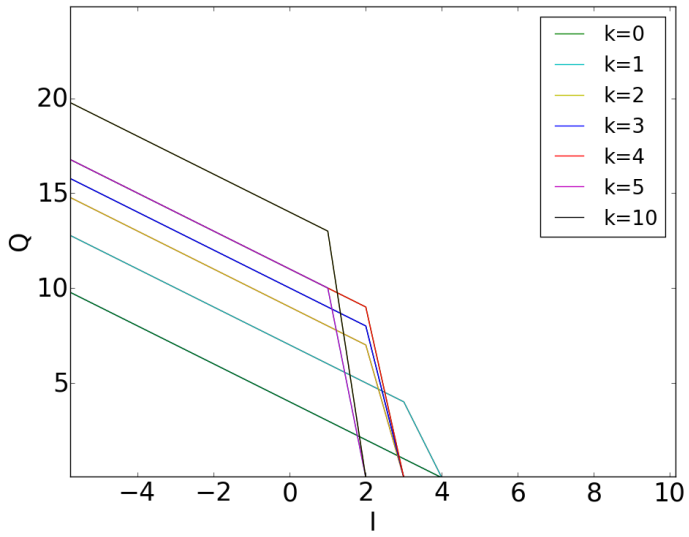


Figure 4.2: Optimal order function W_{op} depending on fixed order costs k.

Simulation

Figure 4.3 shows a comparison between VI and simulation. It is the same bar graph explained in Figure 4.1b, the red line is the frequency of occurrence of the simulation. Both graphs have almost a similar shape, although the red line shows an increase of inventory at an inventory of 13.

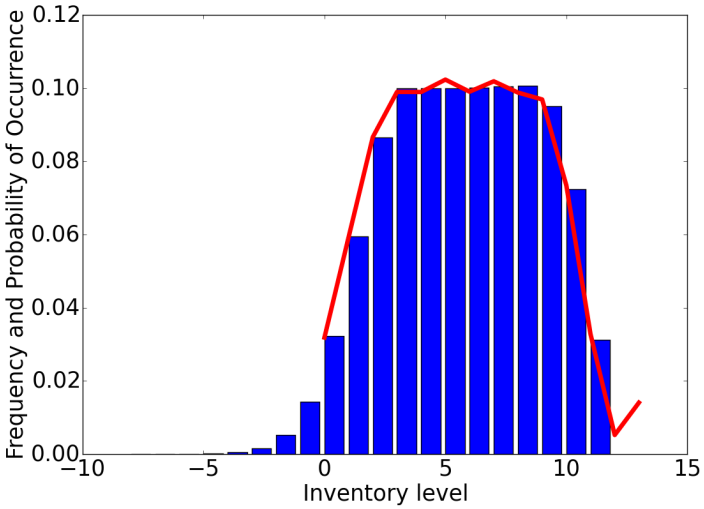


Figure 4.3: Bar diagram of the probability of occurrence to have a certain amount of inventory in stock, as Markov stationary state (blue) and the frequency of inventory level is calculated by the simulation model (red).

4.1.2 Conclusion

The order policy found by Value Iteration dictates that an order is placed when an inventory level equal to the average demand is reached (or is lower than the average demand), but this could also depend on the costs per product and the holding cost. This results in an (s, S)- order policy for non-perishable products (shown in figure 4.1a). If fixed order cost k increases, the moment of ordering is postponed, and I becomes equal to μ (as expected). This means when fixed order costs increase, the re-order level s is as low as possible (the inventory is used completely). Because the costs of back-ordering are slightly higher than pre-ordering. In case of low fixed order costs, the re-order level s is increased up to $2 \times \mu$. In this case, an inventory level of $2 \times \mu$ covers 97% of all demand. When k is small or zero, the difference in costs between back-ordering and pre-ordering Q, becomes bigger. Which makes it less appealing to have low inventory levels. However, non-perishability causes inventory levels to be higher than the expected demand μ. Fixed order costs k do appear to be high, compared to holding costs h.
Theoretically, when fixed cost k is high, the optimal re-order inventory level s could be at a negative inventory level. An order is placed when back-ordering becomes more expensive than ordering a large quantity, although there are fixed costs and inventory costs.
So, when back-ordering is permitted and the inventory is not perishable, an (s, S)-order policy is the optimal policy. Re-order level s depends on the equilibrium between fixed order costs, holding costs and back-order costs.

4.1.3 Discussion

Two types of calculations were performed to find the optimal order policy (by VI), and to evaluate this policy (by simulation). The average costs were the same for both methods. The frequency and probability of occurrence graph (figure 4.3) did show quit similar outcomes, however it did show two deviations. At $I = 6$ and at $I = 12$. At $I = 6$ the simulation, showed a dip, as well $I = 12$. However, at $I = 7$ and $I = 13$ the probability of occurrence did increase. The reason for this behaviour could be the used order-policy, and the equilibrium between back-order costs, holding costs and order costs.

4.2 Non-perishable Products: Case 2

4.2.1 Results

In this case back-ordering is no longer permitted, which means the retailer could have loss of sales. Not only costs are relevant but also the selling price. Table 4.3 shows which input data is used. Table 4.4 shows the average profit for both VI and simulation. The difference δ between the two models is equal to 2.3%.

Table 4.3: Used input data.

μ	2
h	0,25
k	4
f	3

Table 4.4: Average profit of the order policies, when back-ordering is not permitted and sales price is taken into account. Using VI and simulation.

Average Profit	
Value Iteration	3.565
Simulation	3.648
δ	0.083

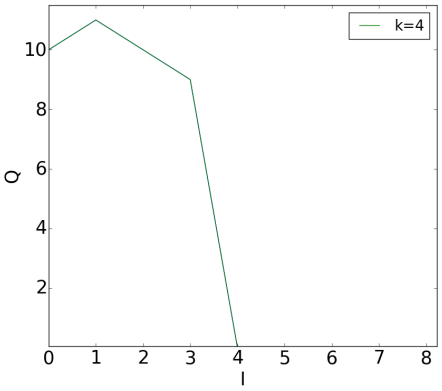
Value Iteration

Due to the fact that re-ordering is not possible, the minimum inventory level in this case is 0 and the maximum inventory level is 12 (because of order-up-to level S).

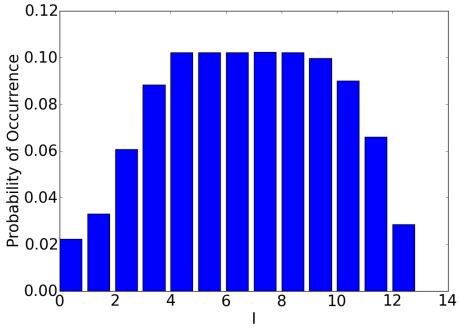
$Q_{op} = 10, 11, 10, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0$

Where $Q_{op}(0) = 10$; $Q_{op}(1) = 11$ etcetera.As depicted in Figure 4.4a below re-order level $s = 4$ an order should be placed. Figure 4.4b shows the occurrence of inventory.

The figure has a peak between 4 and 9 products in stock.



(a) Optimal order quantity $Q_{op}(I)$ as function of begin inventory I. Inventory I could not be negative, because of that the inventory level starts at zero.



(b) Shows the probability of occurrence to have a certain amount of inventory in stock, as Markov stationary state.

Figure 4.4: Optimal order quantity and probability of inventory when performing VI.

Figure 4.5 analyses the order policy when the fixed cost is varied. The higher the fixed costs, the lower the inventory level is, when the order is placed until average demand level is met. If the fixed costs are still too high, the model does not order anything. In this thesis, the fixed cost of $k = 4$ is used for a comparison between cases.

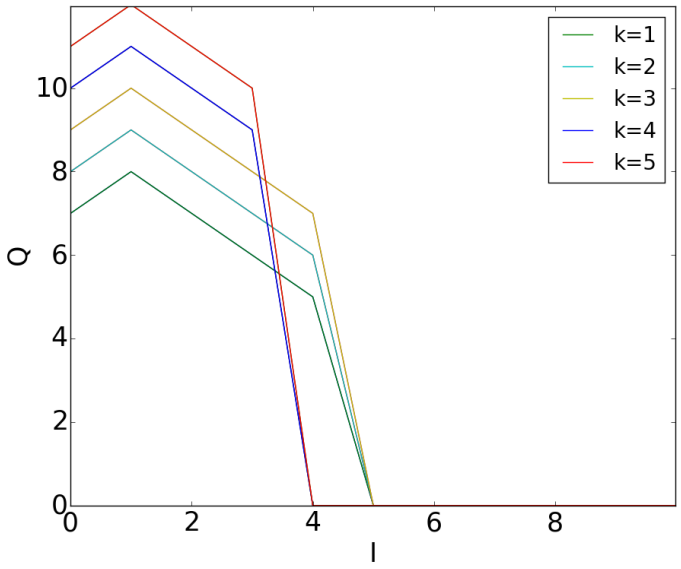


Figure 4.5: Optimal order function W_{op} depending on fixed order costs k.

Simulation

Figure 4.6 shows a graph with the outcome of the Markov matrix (based on the optimal policy) and the simulation. The simulation follows the probability of occurrence based on the Markov stationary state, except for the occurrence of 10 products in the inventory, the occurence of 10 products in stock is higher in the simulation.

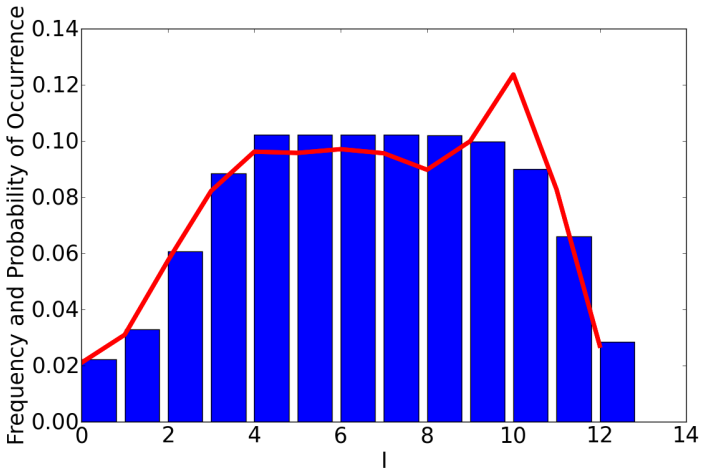


Figure 4.6: The bar diagram the probability of occurrence to have a certain amount of inventory in stock, as Markov stationary state (blue) and the frequency of inventory level is calculated by the simulation model (red).

4.2.2 Conclusion

This case is based on optimizing profit when back-ordering is not allowed. This means lost sales will occur if the inventory level is not sufficient. To fulfil demand, an equilibrium should be found between order costs, holding costs and lost sales. If inventory level $I \geq 1$, the VI results into an (s, S)-order policy. This policy has an order-up-to level S of 10, which means (identical to case 1) a high probability of occurrence for $I = 10 - \mu$ due to the lead-time, or smaller until re-order level s is reached ($s \leq I \leq S - \mu$).
After increasing the fixed costs k, re-order level s does decrease, the same happened in case 1. However, in this case $s \geq 2 \times \mu$, because the chance of lost sales will be too big when $s \leq 2 \times \mu$. In case of decreasing fixed costs k, the re-order level s increases (also identical to case 1).

4.2.3 Discussion

The generated optimal order policy reveals a deviation from the (s, S)-order policy and optimal order policy of case 1, is at $I = 0$. As figure 4.4a shows, the optimal order quantity of $I = 0$ is lower than the optimal order quantity of $I = 1$. This deviation does not meet the inventory control literature of chapter 2. A second deviation, is found in the frequency and probability of occurrence graph (figure 4.6). The simulation, which evaluates the outcome of the VI, has the same average profit. But figure 4.6 shows in the simulation a peak frequency at $I = 10$, $I = 10$ is equal to S in this case. The VI shows a smooth curve following the probability of occurrence of case 1.

4.3 Non-perishable Products: Case 3

4.3.1 Results

Case 3 focusses on fulfilling the customer service level (CSL). The customer service level taken in this case is $\alpha = 0.9$, this means 90% of customer demand should be fulfilled. Again, two calculations were performed: VI and Simulation. The difference δ between the two models is equal to 0.94%.

Table 4.5: Used input data of case 3.

μ	2
h	0,25
k	4
α	0.9

Table 4.6: The average costs of the order policies, when back-ordering is not permitted. For VI and simulation.

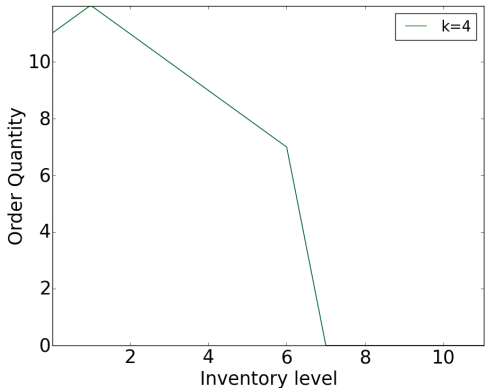
Average Costs	
Value Iteration	2.606
Simulation	2.604
δ	0.0245

Value Iteration
The optimal order quantity Q_{op} per inventory level is shown below. Back-orders are not permitted, which means that the indices of Q_{op} are representing the current inventory level. Value 11 means there are no products in stock ($Q_{op}(0) = 11$), value 12 means there is 1 product in stock ($Q_{op}(1) = 12$), etc.

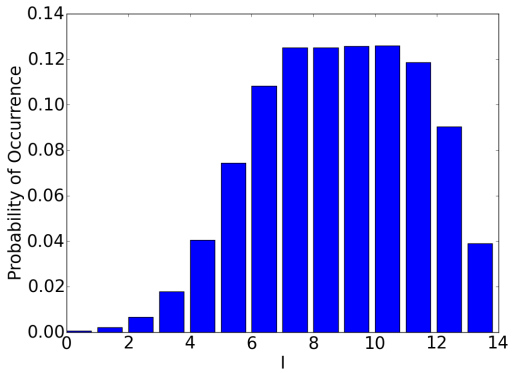
$$Q_{op} = 11, 12, 11, 10, 9, 8, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0$$

Figure 4.7a shows that orders are placed when the current inventory level is equal, or smaller than 6. The inventory level is returned to 13 when ordering, except for inventory level 0. At $I = 0$, the inventory is increased up to 11. Figure 4.7b shows in most of the cases the inventory level is between 6 (s_0) and 11. Which are the inventory levels for when no orders are placed.

The Q_{op} shown above, is an optimal when fixed cost of $k = 4$ is used. Figure 4.8 shows what happens when the fixed cost k values are varied. When the fixed costs $k = 0 - 2$ are used, odd curves are the results, while for $k = 3 - 5$ the same patterns are showed as the fixed costs k of other cases.



(a) The optimal order quantity for varying inventory levels.



(b) Shows the probability of occurrence to have a certain amount of inventory in stock, as Markov stationary state.

Figure 4.7: Optimal order quantity and probability of inventory when performing VI.

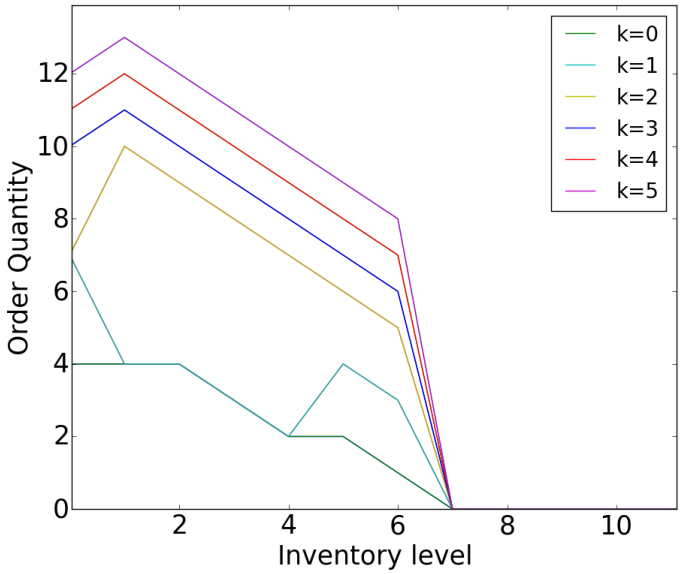


Figure 4.8: Optimal order function Q_{op} depending on fixed order costs k.

Simulation

Figure 4.9 shows nearly the same properties for the Markov stationary state (VI model), and simulation.

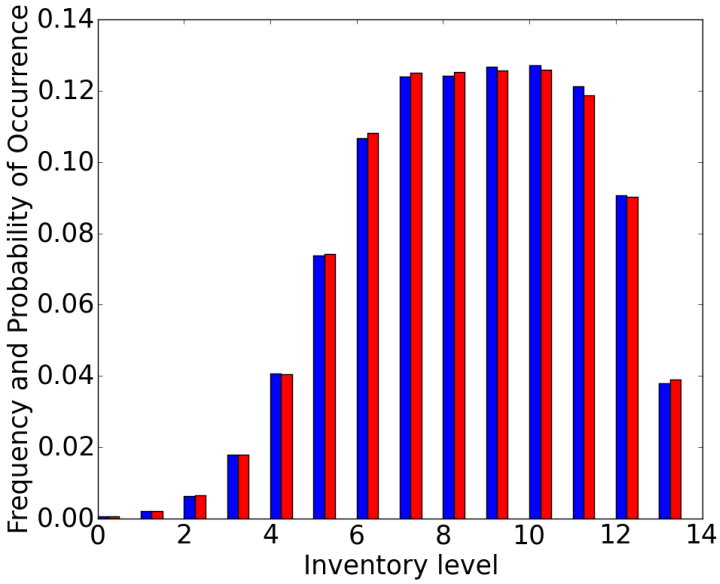


Figure 4.9: Bar diagram of the probability of occurrence to have a certain amount of inventory in stock, as Markov stationary state (blue) and the frequency of inventory level is calculated by the simulation model (red).

4.3.2 Conclusion

Due to the fact that in this case $CSL \geq \alpha$, a minimum inventory level s_0. Which should be kept, to be able to fulfil in 90% of the cases, the customer demand can be derived. For $k \geq 3$, the properties are the same as in case 2: An (s, S)- order policy with a dip at $I = 0$. However, for $k \leq 2$ the optimal order quantity per fixed costs level differentiates from previous cases and higher fixed order cost levels. A possible explanation might be periodic ordering instead of continuous re-order/review policy. Due to the minimum inventory level s_0, for all fixed order costs k, the ordering starts at the s_0. Instead of an increasing re-order level when fixed costs are decreasing. Decreasing re-order level s would mean the CSL might not be met. Increasing the re-order level s on the other hand, causes more expenses, because the costs equilibrium is not met.
The frequency and probability of occurrence graph (figure 4.9), shows a very similar pattern for both stationary state and frequency. Combined with the average costs, it means that the simulation is really close to the VI. The most frequent I-levels is between $s_0 \leq I \leq S - \mu$.

4.3.3 Discussion

The fixed costs $k \leq 2$ are showing odd patterns as optimal order policy Q_{op}. For $k = 1$ and $k = 2$ at $I = 0$, the inventory level goes back to the minimum inventory level s_0.

The pattern might have something to do with periodic ordering, but this is not shown by the model.

4.4 Perishable Products: Case 4

4.4.1 Results

Case 4 differs from the other cases by the fact that the inventory is perishable. The products expire after being held for 1 day. This means two kinds of inventory exist (at the beginning of the day): 0-days old inventory I_0, and 1-day old inventory I_1. When keeping the perishability aside, the case has similarities with case 2. Table 4.7 shows the used input data in this case. However, multiple situations containing varied fixed costs k and varied holding costs h values are simulated in this case.

Table 4.7: Used input data.

c	1
μ	2
h	0,25
k	1
f	3

Table 4.8: The average costs of order policies, when back-ordering is not permitted and selling price is taken into account.

Avg. Profit	$k = 0$ $h = 0.1$	$k = 1$ $h = 0.1$	$k = 0$ $h = 0.25$	$k = 1$ $h = 0.25$	$k = 4$ $h = 0.25$
Value Iteration	3.152	2.259	2.957	2.080	0.0
Simulation	3.157	2.259	2.963	2.088	0.0
$\delta[SIM - VI]$	0.005	0	0.006	0.008	0
$\delta[ratio]$	0.0016	0	0.002	0.0038	0

Value Iteration

The minimum inventory ($\underline{I}$) used for the VI is 0, because back-ordering is not permitted. The maximum inventory is the Order-up-to level S, 5. The obtained optimal order quantity is depicted in Table 4.9, when $k = 1$ and $h = 0.25$. For both I_0 and I_1, the range of inventory is between $\underline{I}$ to S. Table 4.9 is an example of one of the possibilities of an optimal order quantity matrix, in this case the optimal order quantity is based on $k = 1$ and $h = 0.25$. This matrix exists for every k/h -case, depicted in a 3D-graph below.The frequency of occurrence is only calculated in the simulation. Therefore the figures below are based on the optimal order quantity of VI and the frequency is based on the simulation model.

Table 4.9: The optimal order quantity, when I_0 and I_1 have a certain amount of inventory level, ranging from 0 to $s[5]$. For $k = 1$ and $h = 0.25$.

Q_{op}		I_1					
		0	**1**	**2**	**3**	**4**	**5**
	0	3	3	3	3	3	3
	1	3	3	3	3	2	2
	2	3	3	3	0	0	0
	3	3	0	0	0	0	0
I_0	**4**	0	0	0	0	0	0
	5	0	0	0	0	0	0

Simulation

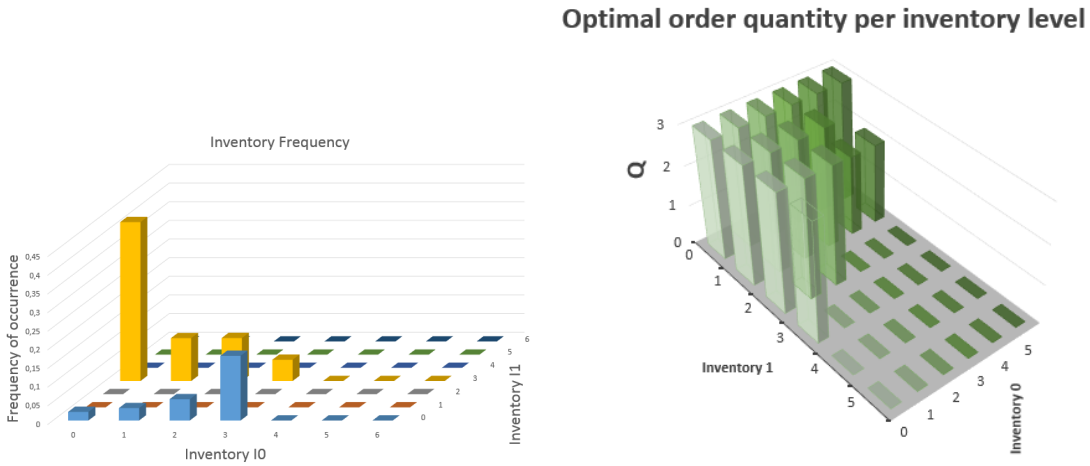
Table 4.8 compares the profit of VI against profit of the simulation, for multiple k and h values. The highest profit is obtained when $k = 0$ and $h = 0.1$. And the lowest profit is obtained when $k = 1$ and $h = 0.25$. Keeping the situation with $k = 4$ and $h = 0.25$ out of scope, when no profit was made. The difference in average profit, shows how close the simulation is to the optimal found after using VI. All differences are $p(\delta) < \alpha$, when $\alpha = 0.05$. To know whether the order policy turns periodic, it is important to check the ordered order quantities. Table 4.10 seems to show only when $k = 0$ and $h = 0.25$, the order policy is not periodic. However, when expanding the array with ordered order quantities, it turns out all orders containing $k \leq 1$ are non-periodic. If $k > 1$, the orders become periodic.

Table 4.10: Order quantities based on time t.

$k = 4, h = 0.25$	q=[0,0,0,..,0,0]
$k = 2, h = 0, 25$	q=[4,0,4,..,0,4]
$k = 1, h = 0.25$	q=[3,3,0,..,0,3]
$k = 1, h = 0.1$	q=[3,0,3,..,3,3]
$k = 0, h = 0.25$	q=[2,3,1,..,2,2]
$k = 0, h = 0.1$	q=[2,2,3,..,2,3]

Figures 9.3a- 9.6b (Appendix), show the outcome of the optimal order quantity of VI, and the frequency as simulation output. Table 4.9 is the optimal order quantity when $k = 1$ and $h = 0.25$, this situation will be explained in this chapter, whereas the other situations are built upon the same strategy.

Figure 4.10a shows which inventory levels have a high frequency of occurrence. The figure shows at $I_0 = 0$ and $I_1 = 3$, a peak in frequency of occurrence. A smaller peak at $I_0 = 3$ and $I_1 = 0$. The first peak has a frequency of 0.425 and the second peak a frequency of 0.173. Whereas figure 4.10b shows the matching optimal order quantity. For both, most common, inventory levels (I_0 and I_1) applies an order quantity of 3, when one (or both) of the inventory levels is zero. This order quantity is also confirmed by the order quantity based on t (Table 4.10).



(a) Frequency of occurrence of inventory I_0 and I_1 ($k = 1$ and $h = 0.25$).

(b) Optimal order quantity when containing two levels of perishability (I_0 and I_1), the inventories could not be negative. For $k = 1$ and $h = 0.25$

Figure 4.10: Frequency and Optimal order quantity for $k = 1$ and $h = 0.25$.

4.4.2 Conclusion

The fixed costs are of great importance in determining the correct order policy. Due to perishable inventory it is not possible to order great amounts in advance to keep the fixed order costs low. For such high fixed costs it is not possible to make profit any more, no orders are placed (in this case when $k \geq 3$). When it is still possible to make a slight profit, the orders become periodic. Which means that orders are made depending on perishability. Which in this case is every other day (when $k = 2$). When the fixed costs are low enough to order independent of the perishability (cost wise), an order policy could be found. Still the order quantities are highly bound to the expected demand. It could differ slightly, because of its one-day perishability but only what the Poisson distribution allows. Changing the holding cost h does not affect the order policy.
It is important to keep the frequency of occurrence graph aside of the optimal order policy figure. To know which orders are most likely to happen, which in this thesis is 0 or 3. However, the sequence of ordering shows it is not periodic.

4.4.3 Discussion

In this case no Markov stationary state was made to compare with the frequency of the simulation. However, the average profits of the VI and the simulation do match.

Chapter 5

Curse of Dimensionality

To tackle the curse of dimensionality it is important to know how to deal with it theoretically. In Chapter 1 the curse of dimensionality was described, this chapter explains how to deal with it and how to set-up correct limits for a VI. Powell (2007) wrote the state space, the outcome space and the action space should be kept small. This also means that the boundaries need to be strict, this prevents calculating infeasible solutions. In this thesis, the curse of dimensionality is mainly be tackled at the state space.

The state space s has to do with all states that a model can have (Kettenis and Van der Vorst, 2007). Both Powell (2007) and Hendriks (2007) recalled the state space as the reason for existing of the curse of dimensionality. Hendriks (2007) specifically states for the Bellman equation, which is used in this thesis, that an increase of dimension also results in an increase of state space. The Bellman equation encounters $\forall s \in \#S : V(s)$, in which $\#S$ is the number of elements in set S. When increasing with dimension n, the calculation time will increase as well. For example, if s could have 10 values, the stored $\#S = 10^n$. This means when the VI has 3 dimensions, the Bellman equation should encounter the calculation $10^3 = 1000$ times in each stage. Besides $\#S$ also $\#X$ influences the calculation time for a Bellman calculation (Hendriks, 2007).
2D is used when the inventory does not have an expiring date, 3D when the product expires after 1 day, 4D after 2 days etcetera. As stated earlier, the amount of calculations required per increase of dimension, increases exponentially. In this thesis, the state space contains inventory and order quantity, which goes up to three dimensions. The Poisson distribution is used to keep the state space of inventory as small as possible. The Poisson distribution is based on the replenishment cycle and two times the average demand μ. Via this way all reasonable states are calculated (according to the distribution) and the chance that the correct value is within the distribution is pre-determined by α. The Economic Order Quantity EOQ was calculated before starting the VI, to have a rough estimation of what the optimal order quantity q can be. This causes a smaller state space of q.

In this thesis the calculation of the VI only happens when the span, difference between V and W, is larger than α. Because the computer is able to keep computing the

difference until infinity. If the program is not running well, or requires too many loops, an ϵ can be introduced in a while-loop. This ϵ causes the model to stop after a specific number of iterations. This is also confirmed by Rust (1997), he states when a problem (in their case a Markov Decision Problem, MDP) is discrete, the problem could be solved exactly. However, when a MDP is continuous, an arbitrary small solution tolerance should be encountered, which is in this thesis α. This is part of the Bellman's curse of dimensionality (Rust, 1997).
Pakes and McGuire (2001) wrote that two causes of the curse of dimensionality are found, 1) in the number of points evaluated at each iteration and 2) the time per point required. The first statement is tackled in this thesis by limiting the following input the average demand μ was set low, to avoid extra calculations. The second statement, the reduction of time per iteration, is tackled by replacing do-loops by minimization and maximization functions. $max(d, 0)$ e.g. instead of a do-loop of multiple lines.

So, according to the references used in the text above, multiple reasons for the curse of dimensionality exist. From state space to time required per iteration, all possible causes of the "curse" were encountered in this thesis, and decreased in size as much as possible.

Chapter 6

General Conclusion

According to the outcomes of the models of Case 1-3, when having non-perishable products, the optimal order policy turns out to be an (s, S)- order policy. This is also in line with what Tekin et al. (2001) said (Chapter 2). The first case is allowed to have a negative inventory level, due to back-ordering. The found order policy follows exactly the behaviour of how Pauls-Worm (2007) did describe an (s, S)- order policy (Chapter 2). The next case has its inventory levels limited by setting the minimum inventory level at $\underline{I} = 0$ and loss of sales is a possibility. Despite of these limitations, the (s, S)- order policy is still optimal, however it shows a dip at inventory level $I = 0$. The same happens when a customer service level is set and should be met. This happens by introducing a minimum re-order level s_0. So, for non-perishable inventory the optimal order policy is an (s, S)- order policy.

For perishable inventory it is more difficult to find an optimal order policy. This policy depends on the amount of fixed costs k, changing the holding cost h did not influence the outcome of the model. High fixed costs cause periodic ordering, but too high fixed costs cause no ordering at all. However, when optimizing the order policy with low fixed costs, an order policy is shown. The inventory levels which are most likely to occur in case 4, are shown in figure 4.10a. These most occurring inventory levels are equal to: $0, \mu$ or $\mu + 0.5\mu$. The values are concluded from the frequency of occurrence graph in combination with the optimal order quantity graph (figure 4.10b). This policy depends on the perishability of the product and the service level put into the distribution.

When comparing the VI with the results of the simulation, it turns out that the simulation comes close to the VI comparing the average costs/profit. After using the optimal order quantity obtained during the VI, it turns out that for all cases, the frequency (simulation) and probability of occurrence (using Markov matrix) are closely related. Which means that the simulation models designed to evaluate the VI, do show the same outcomes. This means that the simulations could be used as model, which means less calculations are necessary per case.

To deal with the curse of dimensionality it is important to limit the state space (most important in this thesis). This curse of dimensionality especially happen when the Bellman equation is used. When increasing the dimension, the number of state elements in the set will increase exponentially. In this thesis, the increase is caused by adding perishability. In case of non-perishability the model is 2D, do the products expire after 2 days, the model turns to 3D, etcetera. The first attempt to keep the state space as small as possible, was by limiting the inventory levels by using the Poisson distribution. The second attempt was to keep the state space of order quantity as small as possible was by first determining the EOQ to know what the global optimal order quantity will be. These two approaches were used to keep the state space as small as possible, however by introducing a span the running time is reduced as well.

To wrap up, the value iteration is based on the Bellman equation. This equation is used in all cases. By changing the aim slightly per case, it was possible to see what happened step-by-step. A general pseudo-code was designed which applied for all cases and functions as back-bone of this research. The value iteration for perishable inventory was more difficult, because another dimension was required. Adding the dimension was the only big difference regarding the calculations of the used pseudo-code. So, value iteration of perishable inventory does have a lot of similarities compared to non-perishable inventory. However, dimensions should be added, which makes it complicated when the time of perishability increases. Complicated in sense of calculation, but also the number of states increase, which could result in problems caused by the curse of dimensionality.

Chapter 7

General Discussion

The optimized costs or profit, depending on the case, are (nearly) similar. This means the simulation is a close evaluation of the value iteration. However, when comparing the frequency of occurrence, obtained by the Markov stationary state, deviations are visible. The probability of occurrence has a well shaped and distributed figure. Whereas the frequency of occurrence does show multiple outliers.
In Chapter 4 all cases were discussed. Case 1 had an increase at the end of the frequency of occurrence graph. Case 2 showed a dip at the inventory level $I = 0$, of the optimal order quantity $Q_{op}(I)$-graph. This is not in line with the (s, S)-order policy. Another deviation in Case 2 was the frequency of occurrence at inventory level $I = 10$ of the frequency of occurrence graph. This deviation could be explained by the fact that the order-up-to level $S = 12$, but the leadtime is 1 day, this means that the expected demand μ should be withdrawn. Case 3 did show unexpected demand patterns for fixed costs lower than 3. Case 4 does not have the probability of occurrence graph based on the Markov matrix. This makes it only possible to compare the outcomes of average profit by the simulation and the value iteration. And to make an estimation of the most frequent ordered order quantities.

Chapter 8

Further Research

For further research the increase of average demand could be included, it is worth knowing what happens with the optimal order quantity and model when larger values are used. Another interesting topic is to find the influence of the fixed costs k, using VI and evaluation by simulation. The fixed costs turn out to be of great influence in this research. It is interesting to know what happens, using different fixed costs k, when the shelf-life of the inventory increases. This will influence the order policy, it could become periodic or even no orders placed.
Another case for further research are the declines at $Q_{op}(0)$, shown in the figures 4.5 and 4.8. These declines seem to be regularly, which means there must be an explanation for this behaviour.
In case 3, the order policies for low fixed costs show large deviations. It is interesting to find out what these deviations mean and where they come from. After checking the model, the orders were not periodic for small fixed costs. However, this assumption should be included in further research.
For all cases applies the simulation is close to the outcomes of the VI. Comparing the frequency of occurrence (simulation) and the probability of occurrence (Markov stationary state of VI), the figures do have many the same characteristics. However, some of the simulations do show strange outliers, case 2 e.g., for further research it is worth knowing why the simulations do show outliers and to check the reliability of these outliers.

A challenge for further research could be the comparison of the value iteration used in this thesis, with the (Q, r, T)-order policy of Tekin et al. (2001). Because it focusses on two triggers, the inventory level s and the shelf-life T. It is interesting to know whether the value iteration and the (Q, r, T)-order policy are close to each other when measuring costs. Both policies do take the shelf-life and the inventory level into account when determining the order quantity.

Bibliography

R. Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, pages 679–684, 1957.

R. Bellman. Dynamic programming. 153(3731):34–37, 1966. doi: 10.1126/science.153.3731.34.

L. Bertazzi, G. Paletta, and M. Speranza. Deterministic order-up-to level policies in an inventory routing problem. *Transportation Science*, 36(1):119–132, 2002.

R. A. Broekmeulen and K. H. Van Donselaar. A heuristic to manage perishable inventory with batch ordering, positive lead-times, and time-varying demand. *Computers & Operations Research*, 36(11):3013–3018, 2009.

S. Chopra and P. Meindl. *Supply Chain Management: Strategy, Planning, and Operation*, volume 6. 2016.

EC. Preparatory study on food waste across eu 27, 2010.

M. Eriksson. *Retail Food Wastage*. PhD thesis, 12 2012.

FAO. Save food: Global initiative on food loss and waste reduction, 2011. URL `http://www.fao.org/docrep/014/mb060e/mb060e.pdf`.

S. Fawcett, L. Ellram, and J. Ogden. *Supply Chain Management From Vision to Implementation*. Pearson Education Limited, 2014.

H. Fischer. *History of the central limit theorem : from classical to modern probability theory*, chapter Markov Processes, pages 19–27. Springer, 2011a.

H. Fischer. *History of the central limit theorem : from classical to modern probability theory*, chapter Poisson Processes, pages 17–19. Springer, 2011b.

R. Haijema. A new class of stock-level dependent ordering policies for perishables with a short maximum shelf life. *International Journal of Production Economics*, 143(2):434 – 439, 2013. ISSN 0925-5273. Focusing on Inventories: Research and Applications.

R. Haijema and S. Minner. Stock-level dependent ordering of perishables: A comparison of hybrid base-stock and constant order policies. *International Journal of Production Economics*, 181:215 – 225, 2016. ISSN 0925-5273. SI: ISIR 2014.

T. Hendriks. Inventory management. In G. Claassen, T. Hendriks, and E. Hendrix, editors, *Decision Science: Theory and Applications*, chapter 8, pages 181–183. Wageningen Acad. Publ., 2007.

E. Hendrix, G. Ortega, R. Haijema, M. Buisman, and I. Garcia. On computing optimal policies in perishable inventory control using value iteration. 2018.

E. Hendrix, C. Kortenhorst, and G. Ortega. On computational procedures for value iteration in inventory control. *IFAC*, 2019.

R. M. Hill and S. G. Johansen. Optimal and near-optimal policies for lost sales inventory models with at most one replenishment order outstanding. *European journal of operational research*, 169(1):111–132, 2006.

D. Kettenis and J. Van der Vorst. Inventory management. In G. Claassen, T. Hendriks, and E. Hendrix, editors, *Decision Science: Theory and Applications*, chapter 13, page 344. Wageningen Acad. Publ., 2007.

S. Nahmias. *Perishable inventory systems*, volume 160. Springer Science & Business Media, 2011.

A. Pakes and P. McGuire. Stochastic algorithms, symmetric markov perfect equilibrium, and the 'curse'of dimensionality. *Econometrica*, 69(5):1261–1281, 2001.

J. Parfitt, M. Barthel, and S. Macnaughton. Food waste within food supply chains: quantification and potential for change to 2050. *Philosophical transactions of the royal society B: biological sciences*, 365(1554):3065–3081, 2010.

K. Pauls-Worm. Inventory management. In G. Claassen, T. Hendriks, and E. Hendrix, editors, *Decision Science: Theory and Applications*, chapter 11, pages 257–284. Wageningen Acad. Publ., 2007.

V. Powell and L. Lehe. Markov chains. URL `http://setosa.io/ev/markov-chains/`.

W. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

D. Prak, R. Teunter, and J. Riezebos. Periodic review and continuous ordering. *European Journal of Operational Research*, 242(3):820–827, 2015.

Prasanth. Simple statistics with scipy, 2011. URL `https://oneau.wordpress.com/2011/02/28/simple-statistics-with-scipy/`.

J. Rust. Using randomization to break the curse of dimensionality. *Econometrica: Journal of the Econometric Society*, pages 487–516, 1997.

E. Tekin, Ü. Gürler, and E. Berk. Age-based vs. stock level control policies for a perishable inventory system. *European Journal of Operational Research*, 134(2):309–329, 2001.

H. Tijms and H. Groenevelt. Simple approximations for the reorder point in periodic and continuous review (s, s) inventory systems with service level constraints. *European Journal of Operational Research*, 17(2):175 – 190, 1984. ISSN 0377-2217.

K. Van Donselaar, T. van Woensel, R. Broekmeulen, and J. Fransoo. Inventory control of perishables in supermarkets. *International Journal of Production Economics*, 104 (2):462–472, 2006.

M. Verleysen and D. François. The curse of dimensionality in data mining and time series prediction. In *International Work-Conference on Artificial Neural Networks*, pages 761–767. Springer, 2005.

Chapter 9

Appendix

9.1 List of Symbols

Chapter 1	
Re-Order Point	s
Re-Order Point	ROP
Total order quantity	Q
Periodic review	R
Order-up-to level	S
Value Iteration	VI
Value function over state s	V_s
State	s
Previous value function over state s	W_s
Contribution of state s and action x	$C(s,x)$
Expected contribution of state s and action x and expectation e	$EC(s,x,e)$
Transition of state s and action x	$T(s,x)$
Alpha service level	α
Inventory level	I
Order quantity	q
Probability	P
Chapter 2	
Base stock policy	BSP
Constant order policy	COP
Lead time	t_l
Minimum order quantity	$\underline{q}$ or q_{min}
Maximum order quantity	$\overline{Q}$ or Q_{max}
Chapter 3	
Demand	d
Average demand	μ
Begin inventory	I_b
End inventory	I_{end}
Optimal order quantity	Q_{op}
Holding cost	h
Fixed order cost	k
Back-order cost	c
Valuation calculation per order quantity	v_s
Maximum difference for optimum	ϵ
Sales price	f
Customer service level	CSL
Probability matrix for CSL	P_{end}
Probability of i to j	ψ_{ij}
Time in all T-periods	$t \in T$
Inventory of 0-days old	I_0
Inventory of 1-days old	I_1
Difference	δ

9.2 Extra Knowledge

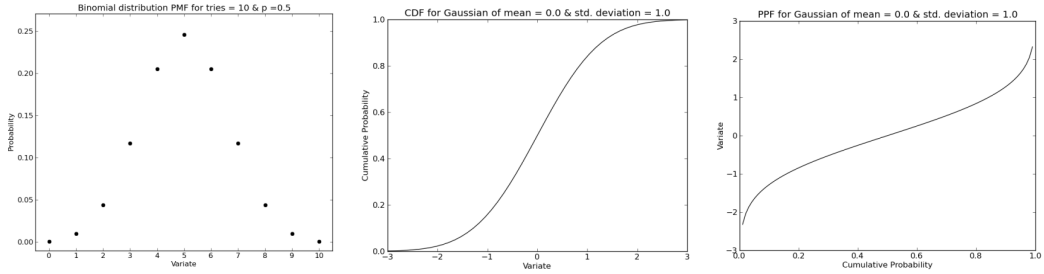
This Chapter has as function to explain parts of mathematical theories used in the models. Of whom I needed more information to understand what the theory would encounter.

9.2.1 Poisson Distribution

According to Fischer (2011b), the Poisson distribution encloses random occurring situations. For example customers which arrive in a store, completely independent of each other. Therefore it is important that the number of points in the non-overlapping intervals are independent. Formula 9.1 shows the Poisson formula, $\mathbb{P}$ is the probability of occurrence of what is between the brackets, $t \in T_n$ is the random time at which n number of events happen, $n \in N_t$ is the number of events happening at time t, λ time rate for events to happen,

$$\mathbb{P}(N_t = k) = \frac{\exp^{-\lambda t}(\lambda t)^k}{k!}. \tag{9.1}$$

The Poisson distribution can be divided into four functions (Prasanth, 2011), shown in figure 9.1:
- *Probability density function:*
The probability density function (.pdf in Python), is proportional to the probability of the variate being in a small interval about the given value. And contains continuous variates.
- *Probability mass function:*
The probability mass function (.pmf in Python) contains discrete variates, figure 9.1a.
- *Cumulative density function:*
The cumulative density function (.cdf in Python) gives the probability that the variate has a value less than or equal to the given value. This means that .cdf is cumulative. Could be used for both continuous and discrete variates, figure 9.1b.
- *Percent point function:*
Percent point function (.ppf in Python) is the inverse of .cdf. It gives the value of the variate for which the cumulative probability has the given value, figure 9.1c.



(a) .pmf function, is the same as the .pdf, although .pmf is discrete and .pdf is continuous. The graph shows a distribution and shows a ratio beneath 1.

(b) .cdf function, the function cumulates its distribution up to 1. Continuous or discrete.

(c) .ppf is the inverse of .cdf.

Figure 9.1: The different Poisson distribution styles drawn in graphs. The most important of these graphs is to see the shape.

9.2.2 Markov Decision Process

Bellman (1957) wrote about the Markovian Decision Process (MDP), which is a useful tool for calculating probabilities. MDP is memoryless and calculates the probability of ending in a next state. This calculation is based on a memoryless approach. An example: There are two locations, A and B. A person is currently at location A, and could stay in A, go to B. After this first step, the person could stay in A, could go to B, or could stay in B etcetera. Figure 9.2 shows the possible paths and the MDP matrix shows the ratios of the probabilities. The sum of the columns are equal to 1, which means that for column 1 e.g., which represents location A, the total probability of doing something is 1.0. This is a small example of a Markov matrix. The example is based on the web-page of Powell and Lehe and confirmed by the theory of Fischer (2011a). Which could be used on large scale cases, to determine the probability of occurrence.

$$MDP : (\begin{bmatrix} A \longrightarrow A & B \longrightarrow A \\ A \longrightarrow B & B \longrightarrow B \end{bmatrix} = \begin{bmatrix} 0.6 & 0.2 \\ 0.4 & 0.8 \end{bmatrix})$$

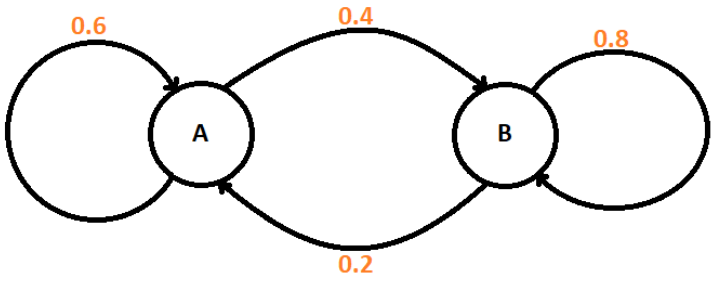


Figure 9.2: Possible paths when moving from state A to B, or remaining in the same state.

9.3 Algorithms

Non-Perishable Products

Case 1

Algorithm 3 Value Iteration for Case 1, in which c,μ, h and k are set as fixed variable.

Set the EOQ to know the $\overline{q}$
Use Poisson.ppf to obtain $\underline{d}$ and $\overline{d}$
To state the probability of demand (p_d) use poisson.pmf (used in case of discrete functions)
pd should be divided by $\sum pd$ to have the right ratio
$\underline{I}$ is equal to negative $\overline{d}$ to know the maximum amount of back-order
$\overline{I}$ is equal to maximum order quantity $\overline{q}$
To compute valuation vector W:
for $i = 0 \dots I$ **do**
    $W_i = k$; first costs are equal to order costs
    for $d = 0 \dots D$ **do**
        $I_{end} = I_i - D_d$
        $W_i = W_i + pd_d$ ($I^+_{end}h$ $-I^+_{end}c$)
while $Span > 0.0001$ **do**
    Slope = $W_1 - W_0$
    for $i = 0 \dots I$ **do**
        valQvec gives the cost vector of all Q's
        for qq=0 ... valQvec **do**
            if $qq > 0$ **then**
                After an order, ordercost should be added: valQvec$_{qq}$=k
            for d=0 ... D **do**
                Inventory when shop closes: $I_{end} = I_i - D_d$
                Inventory after replenishment: $I_{next} = I_{end} + qq$
                if $I_{next} < \underline{I}$ **then**
                    Value = W_0 + ($I_{next} - \underline{I}$)slope
                else
                    Value = W[int($I_{next} - \underline{I}$)]
                $valQvec_{qq} = valQvec_{qq} + pd_d$ (Value+ $I^+_{end}h$ $-I^+_{end}c$)
        V_i= min($valQvec$)
        $opQ_i = int(\arg\min(valQvec))$
    To check whether V_i is optimal or not: Span= max(V-W) - min(V-W)
    avcost = $V_{\underline{I}}$ - $W_{\underline{I}}$

Algorithm 4 Simulation for Case 1.

1: $T = 50000$
2: $\overline{slope} = opQ_{\overline{I}-\underline{I}} - opQ_{\overline{I}-\underline{I}-1}$
3: $\underline{slope} = opQ_1 - opQ_0$
4: **for** t=0 ... T **do**
5: **if** $I_b > \overline{I}$ **then**
6: order = $opQ_{nrstates-1} + \overline{slope}(I_b - \overline{I})$
7: **else**
8: **if** $I_b < \underline{I}$ **then**
9: $order = opQ_0 + \underline{slope}(I_b - \underline{I})$
10: **else**
11: $order = opQ_{I_b - \underline{I}}$
12: **if** $order > 0$ **then**
13: $avcost = avcost + \frac{k}{T}$
14: $Demand = poisson(\mu)$
15: $Inv = I_b - Demand$
16: $Inventory_p = Inv^+$
17: $Inventory_m = (-Inv)^+$
18: $avcost = avcost + \frac{(voor_p h + voor_m c)}{T}$
19: $I_b = integer(Inv + order)$

Case 2

Algorithm 5 Value Iteration for Case 2, in which μ, h, k and f are set as fixed variable. Inventory could be kept without restrictions.

Use Poisson.ppf to obtain $\overline{d}$, $\underline{d} = 0$
D $= 0 \dots \overline{d}$
For Order-up-to level S use poisson.ppf(replenishment cycle$*\mu$)
Create empty vectors for $V, W, Q_{optimal}$
while $Span > \epsilon$ **do**
    $Vfut_0 = W_0$
    for $q = 0 \dots S + 1$ **do**
        $Vfut_q = W_q - k(q > 0)$
    $V_0 = \max(Vfut)$
    $Q_{op_0} = argmax(Vfut)$
    for $i = 1 \dots S + 1$ **do**
        $Vfut = zeros(S - i + 1) - k$
        $Vfut_0 = Vfut_0 + k$
        for $q = 0 \dots S - i + 1$ **do**
            for $d = 0 \dots \overline{d}$ **do**
                $Iend = (i - d)^+$
                $sell = (d, i)^-$
                $Inext = (i - d + q, 0)^+$
                $Vfut_q = Vfut_q + P_d(W_{Inext} + fsell - hIend)$
        $V_i = \max(Vfut)$
        $Q_{op} = argmax(Vfut)$
    $Span = \max(V - W) - \min(V - W)$
Average **Profit** $= V_0 - W_0$

Algorithm 6 Simulation model for Case 2, in which μ, h, k and f are set as fixed variable. Inventory could be kept without restrictions.

T=20000
Import Q_{op} vector from VI case 2
$\overline{q} = max(Q_{opM})$
Set an empty q-vector up-to T
for $t = 0 \dots Time$ **do**
    $Ifreq_b = Ifreq_b + 1$
    demand (d) poisson randomly divided, using μ
    $q_t = Q_{opM_b}$
    if $q_t > 0$ **then**
        ordercost = ordercost + k
    $I = (I_b - d)^+$
    $sell = (d, I_b)^-$
    $sales = sales + sell$
    invcost=invcost+$\frac{hI}{T}$
    $I_b = I + q_t$
$Ifreq = \frac{Ifreq}{T}$
$averagesell = \frac{fsales}{T}$
$averageordercost = \frac{ordercost}{T}$
Average **Profit**=average sell - average ordercost - inventory cost

Case 3

Algorithm 7 Value iteration for Case 3, in which μ, h, k and α-service level are set as fixed variable.

Use Poisson.ppf to obtain $\overline{d}$, $\underline{d} = 0$
D $= 0 \dots \overline{d}$
$EOQ = \sqrt{\frac{2k\mu}{h}}$
For Order-up-to level S use poisson.ppf(α,2μ)+EOQ
minimum order quantity (s_0): poisson.ppf(α, μ)
when ordering is not obliged ($\bar{i}$): poisson.ppf($\alpha, 2\mu$)
$P_{end} = zeros(\bar{i}, \bar{i})$
for $i = 0 \dots \bar{i}$ **do**
    $P_{end_{i0}} = 1 - poisson.cdf(i-1, \mu)$
    for $j = 1 \dots i$ **do**
        $P_{end_{ij}} = P_{i-j}$
for $r = 0 \dots S$ **do**
    $S_r = (s_0 - r)^+$
for $i = 1 \dots \bar{i}$ **do**
    while $chance < \alpha$ **do**
        for $end = 0 \dots i$ **do**
            chance $= chance + poisson.cdf(end + s_i, \mu) P_{end_{i,end}}$
        if $chance < \alpha$ **then**
            $s_i = s_i + 1$
nrstates = S+1
Create empty vectors V, W, Q_{op}
while $Span > \epsilon$ **do**
    $Vfut = zeros(S - s_0 + 1)$
    for $q = s_0 \dots S + 1$ **do**
        $Vfut_{q-s_0} = W_q + k$
    $V_0 = \min(Vfut)$
    $Q_{op0} = argmin(Vfut) + s_0$
    for $i = 1 \dots nrstates$ **do**
        $Vfut = zeros(S - i + 1 - s_i, 0)^+$
        for $q = s_i \dots S - i + 1$ **do**
            if q **then**
                $Vfut_{q-s_i} = Vfut_{q-s_i} + k$
            for $d = 0 \dots \overline{d}$ **do**
                $Iend = (i - d)^+$
                $Inext = (i - d + q, 0)^+$
                $Vfut_{q-s_i} = Vfut_{q-s_i} + P_d(W_{Inext} - hIend)$
        $V_i = \max(Vfut)$
        $Q_{op} = argmax(Vfut) + s_i$
    $Span = \max(V - W) - \min(V - W)$
Average **Profit** $= V_0 - W_0$

Algorithm 8 Simulation for Case 3, in which μ, h, and k are set as fixed variable.

T=20000
$Q = 9$ and $s = 2$
Import Q_{op} vector from VI case 3
for $t = 0 \dots Time$ **do**
    $Ifreq_b = Ifreq_b + 1$
    demand (d) poisson randomly divided, using μ
    $q_t = Q_{opM_b}$
    SL-count = SL-count + $(d \leq I_b)$
    $I = (I_b - d, 0)^+$
    Inventory costs = $inventorycosts + \frac{hI}{T}$
    $I_b = I + q_t$
$Ifreq = \frac{Ifreq}{T}$
Service level = $\frac{SL-count}{T}$
$averageordercost = \frac{k \sum q>0}{T}$
Average Costs = Inventory costs + Order costs

Perishable Products

Case 4

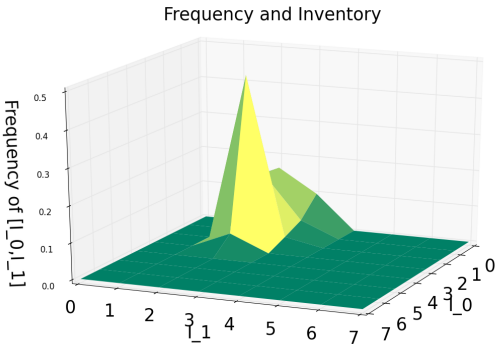
Algorithm 9 Value Iteration for Case 4, in which c,μ, h, k and f are set as fixed variable. Inventory could be kept for 1 day.

Use Poisson.ppf to obtain $\overline{d}$, $\underline{d} = 0$
D $= 0 \dots \overline{d}$
Calculate Newsvendor: $\frac{f-c}{f}$
Maximum Inventory level per I: use poisson.ppf based on Newsvendor and 2μ
Create empty matrices for $V, W, Q_{optimal}$
Create empty vector for Expected Sales, $Esale$
for $i = 1 \dots \overline{d} + 1$ **do**
    $Esale_i = i(1 - poisson.cdf(i, \mu))$
    for $d = 1 \dots i + 1$ **do**
        $Esale_i = Esale_i + dP_d$
$Esale = fEsale$
while $Span > 0.0001$ **do**
    for $i_1 = 0 \dots S + 1$ **do**
        for $q = 0 \dots S + 1$ **do**
            $Vfut_q = W_{q0} - cq - k(q > 0)$
        $V_{0i_1} = maxVfut + Esale_{i1} - hi_1$
        $Q_{op_{0i1}} = argmax(Vfut)$
    for $i_0 = 1 \dots S + 1$ **do**
        for $i_1 = 1 \dots S + 1$ **do**
            for $q = 0 \dots S + 1$ **do**
                Inventory $= (i_0 + i_1, \overline{d})^-$
                $Vfut_q = Esale_{Inventory} - cq - k(q > 0) - hi_1$
                for $d = 0 \dots \overline{d}$ **do**
                    $Inext_0 = q$
                    $overdemand = (d - i_1)^+$
                    $Inext_1 = (i_0 - overdemand)^+$
                    $Vfut_q = Vfut_q + P_d(W_{Inext_0, Inext_1})$
            $V_{i_0 i_1} = maxVfut$
            $Q_{op_{0i1}} = argmax(Vfut)$
    $Span = max(V - W) - min(V - W)$
Average Profit $= V_{00} - W_{00}$

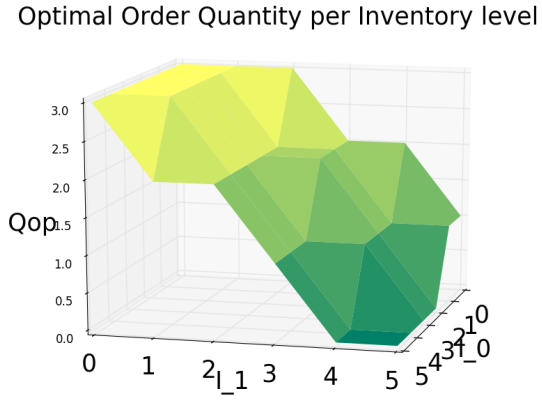
Algorithm 10 Simulation model for Case 4, in which c,μ, h, k and f are set as fixed variable. Inventory could be kept for 1 day.

Use Poisson.ppf to obtain $\overline{d}$, $\underline{d} = 0$
D $= 0 \dots \overline{d}$
Calculate Newsvendor: $\dfrac{f-c}{f}$
Replenishment cycle is based on EOQ: $\dfrac{2k}{h\mu}$
Maximum Inventory level per I: use poisson.ppf based on Repl.cycle and $2 * \mu$
Create an empty matrix for Q_{opM}
for $s = 0 \dots S+1$ **do**
    for $z = 0 \dots S+1$ **do**
        Inv $= (s+z, S)^-$
        Import Q_{op} matrix from VI case 4
$\overline{q} = \max(Q_{opM})$
$\overline{I} = S+1$
Range of I values $= 0 \dots \overline{I}+1$
Create empty vectors (for I_0, I_1) to connect inventory frequencies
for $t = 0 \dots Time$ **do**
    $Ifreq_{0_{I0}} = Ifreq_{0_{I0}} + 1$
    $Ifreq_{1_{I1}} = Ifreq_{1_{I1}} + 1$
    demand (d) poisson randomly divided, using μ
    sell $= (d, I_0 + I_1)^-$
    sales = sales + sell
    shortage = shortage+(d-$(I_0 + I_1)$)$^+$
    $q_t = Q_{opM_{I_0 I_1}}$
    if $q_t > 0$ **then**
        ordercost = ordercost + k
    overdemand=$(d - I_1)^+$
    waste = waste $+(I_1 - d)^+$
    $I_1 = (I_1 - overdemand)^+$
    $I_0 = q_t$
    invcost=invcost+$\frac{hI_1}{T}$
$Ifreq_0 = \frac{Ifreq_0}{T}$
$Ifreq_1 = \frac{Ifreq_1}{T}$
$averagesell = \frac{fsales}{T}$
$averageordercost = \frac{ordercost}{T}$
$averagepurchasecost = \frac{\sum_{q=0}^{S+1} q*c}{T}$
average profit=average sell - average ordercost - inventory cost- average purchasecost
waste percentage $= \frac{waste}{\sum q}$

9.4 Results several k/h-values Case 4

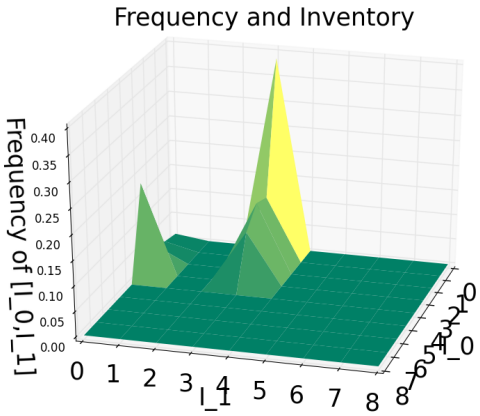


(a) Frequency of occurrence of inventory I_0 and I_1 ($k = 0$ and $h = 0.1$).



(b) Optimal order quantity when containing two levels of perishability (I_0 and I_1), the inventories could not be negative. For $k = 0$ and $h = 0.1$.

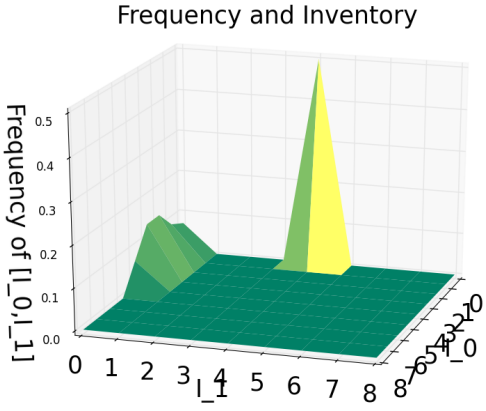
Figure 9.3: Frequency (a) and Optimal order quantity (b) for $k = 0$ and $h = 0.1$.



(a) Frequency of occurrence of inventory I_0 and I_1 ($k = 1$ and $h = 0.1$).

(b) Optimal order quantity when containing two levels of perishability (I_0 and I_1), the inventories could not be negative. For $k = 1$ and $h = 0.1$.

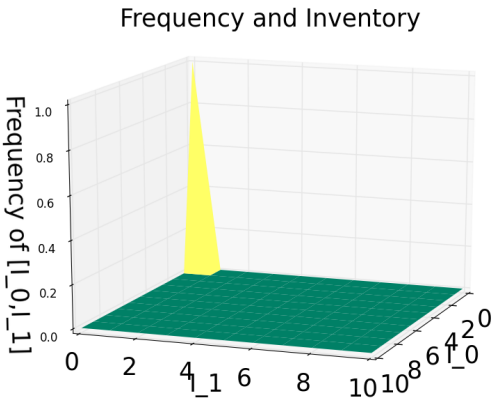
Figure 9.4: Frequency (a) and Optimal order quantity (b) for $k = 1$ and $h = 0.1$.



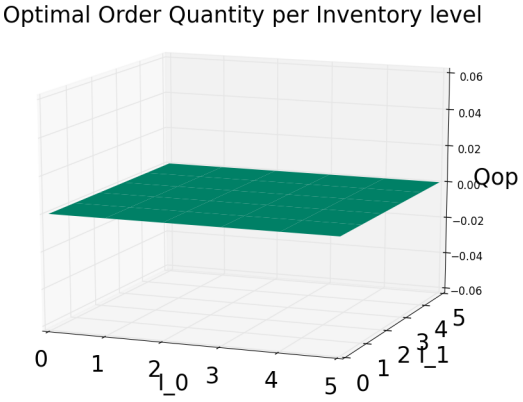
(a) Frequency of occurrence of inventory I_0 and I_1 ($k = 1$ and $h = 0.25$).

(b) Optimal order quantity when containing two levels of perishability (I_0 and I_1), the inventories could not be negative. For $k = 1$ and $h = 0.25$

Figure 9.5: Frequency and Optimal order quantity for $k = 2$ and $h = 0.25$.



(a) Frequency of occurrence of inventory I_0 and I_1 ($k = 4$ and $h = 0.25$)



(b) Optimal order quantity when containing two levels of perishability (I_0 and I_1), the inventories could not be negative. For $k = 4$ and $h = 0.25$

Figure 9.6: Frequency and Optimal order quantity for $k = 4$ and $h = 0.25$.