*Proceedings of the 2012 Winter Simulation Conference*
*C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, eds.*

# SIMULATION TO DISCOVER STRUCTURE IN OPTIMAL DYNAMIC CONTROL POLICIES

Rene Haijema

Wageningen University
Operations Research and Logistics group
Hollandseweg 1, 6706KN, Wageningen
THE NETHERLANDS

Eligius M.T. Hendrix

University of Málaga
Arquitectura de Computadores
Escuela de Ingenerías, c/ Dr Ortiz Ramos
29071, Málaga, SPAIN

Diana van Dijk

Wageningen University, Environmental
Economics and Natural Resources group
Hollandseweg 1, 6706KN, Wageningen
THE NETHERLANDS

Jan van der Wal

University of Amsterdam
Quantitative Economics
Roetersstraat 11, 1018WB, Amsterdam
THE NETHERLANDS

## ABSTRACT

Simulation is known to be a powerful technique to analyze the dynamic behavior of a variety of systems. Dynamic programming is a technique to solve sequential decision making problems. The numerical solution of a dynamic program is a table with an optimal decision for all feasible states of the system. Such a table is usually complex as it lists all states, including those that are not so likely to be visited. In the paper we show how simulation helps in solving the problem efficiently and effectively. For cases on fishery management and inventory control, we show that the combination of simulation and dynamic programming results in a better understanding of the structure of an optimal policy. The insights obtained from simulation help in bounding the state space to speed up the solution process, and have resulted in the discovery a new class of ordering policies.

## 1 INTRODUCTION

Many optimization problems arising in practice involve sequential decision making and can be formulated as dynamic programs, which can be solved by dynamic programming. A special class of dynamic programs is the class of Markov decision problems (MDPs). An MDP formulates a recurring stationary decision problem in a system, for taking actions that optimize some objective function. The objective function must be separable in contributions per period, e.g. the value of the objective function is the (discounted) sum of the contributions per period. Contributions usually are expected costs and/or rewards over a period.

An optimal solution of an MDP, often called an optimal strategy or an optimal policy, is a function that prescribes a best action to take for every state that the system may be in. The number of feasible states can be very large or even infinite. To allow numerical computation of a best action to take in each state, the state space often has to be made discrete by constructing a grid of the state space, which may be truncated to speed up the computational procedure. The solution of an MDP is then a large table of optimal actions. As the number of states on the grid is usually quite large it may be hard to discover structure in such a table. We show in this paper for two cases how simulation can help in setting a grid and in getting a better understanding of the optimal policy. These insights obtained by simulation may help in speeding up the solution process as well as in finding approximate policies that are easier to optimize or to implement.

**Outline** In Section 2, we introduce the framework of Markov decision problems and demonstrate how simulation can help in solving them. The next two sections (3 and 4) present two cases to illustrate

how simulation is used in solving and analyzing MDPs. In Section 5, we discuss more approaches of using simulation in the context of dynamic programs and we conclude the paper.

## 2 HOW SIMULATION MAY HELP SOLVING MARKOV DECISION PROBLEMS

### 2.1 Formulating and Solving (M)DPs

A number of text books have appeared on Markov decision problems (MDPs) and the more general subject of dynamic programs (DPs), see e.g., Bellman 1957; Howard 1960; Puterman 1994. We refer to those books for a detailed description of the framework and solution techniques. In this section, we discuss the main ingredients and point out where simulation can be put into place to help solving MDPs.

Let us first describe the dynamics in infinite horizon MDP in discrete time. At the start of every period, the state of the system, denoted by $\mathbf{x}$, is observed and one decides about the action $a$ to implement. Notice that the state space may be unbounded and continuous. However, to solve the problem numerically, it is made finite and discrete by setting up a grid $\mathscr{X}$ for the state space. As a result of this decision and some uncertain events, the state of the system changes after one period into state $\mathbf{y}$ with a known probability $P(\mathbf{y}|\mathbf{x},a)$. The states $\mathbf{y}$ that can be reached with a positive probability from state $\mathbf{x}$ when taking action $a$ are listed in the subset $\mathscr{X}(\mathbf{x},a)$.

A direct contribution function is associated to a state transition accounting for the costs and or rewards to incur over the period. Weighting these contributions by the transition probabilities $P(\mathbf{y}|\mathbf{x},a)$, provides an expected contribution $\mathbb{E}C(\mathbf{x},a)$ to account for when starting the period in state $\mathbf{x}$ and taking action $a$. Given state $\mathbf{x} \in \mathscr{X}$, the key question is to select action $a$ from action space $\mathscr{A}(\mathbf{x})$ such that the sum of discounted expected contributions over an infinite horizon is maximized or minimized. Alternatively, one optimizes the long-run average (non-discounted) expected contribution over an infinite horizon.

In solving the MDP by value iteration, a so-called value function $V_n(\mathbf{x})$ is used representing the sum of the expected contributions over an horizon of $n$ periods, starting in state $\mathbf{x}$. Given the fact that the objective function is separable and Bellman's principle of optimality holds, the $n$-period problem can be decomposed into a one-period problem and an $(n-1)$-period problem:

$$V_n(\mathbf{x}) = \min_{a \in \mathscr{A}(\mathbf{x})} [\, \mathbb{E}C(\mathbf{x},a) + \sum_{\mathbf{y} \in \mathscr{X}(\mathbf{x},a)} P(\mathbf{y}|\mathbf{x},a)\, V_{n-1}(\mathbf{y}) \,]. \tag{1}$$

One solves (1) for all states $\mathbf{x} \in \mathscr{X}$ starting with $n = 1$, where usually $V_0(\mathbf{y})$ is set to zero. Next $V_n$ is computed for increasing values of $n \in \{1,2,3,\ldots\}$ until the span (=difference between the maximal value and the minimal value) of the vector $V_n - V_{n-1}$ is smaller than a prespecified accuracy $\varepsilon$. The action space may be unbounded and continuous and can be considered as such as long as one has a procedure that is fast enough to solve the optimization problem. This technique is called value iteration and is in fact a dynamic programming approach to solve the problem backwards starting with a one-period problem and then expanding the problem. Other techniques for solving MDPs are policy iteration and linear programming, for which we refer to the earlier mentioned textbooks.

### 2.2 Simulation to Discover Structure

Before as well as after solving an MDP, simulation can be of help to better understand the solution of an MDP. In this subsection, we sketch the general ideas that are illustrated further in the exemplary cases presented in the next two sections. First, before solving an MDP, a grid of the state space needs to be set. By simulation of a reasonable but suboptimal policy one learns about boundaries of the state space. This helps speeding up the solution process significantly; the less states for which we need to evaluate (1), the faster the solution process.

Second, after having solved a (small scale) MDP, one may simulate the thus obtained optimal policy to check whether the assumptions made regarding the state space are valid. Finally, simulation of the optimal policy may reveal whether the optimal policy can be approximated by a simplified policy that does not

require a detailed state description. In the remainder of this paper, these three roles of Simulation in solving MDPs are illustrated.

## 3 CASE 1: PERISHABLE INVENTORY MANAGEMENT (PIM)

In literature, many inventory models can be found that are developed for non-perishable products. The control of inventories of perishable products with a fixed shelf life is less studied and much more complicated. For an overview on models for PIM see (Nahmias 1982; Karaesmen, Scheller-Wolf, and Deniz 2011). In (Haijema, van der Wal, and van Dijk 2007; Haijema, van Dijk, van der Wal, and Smit Sibinga 2009) an MDP and DP approach is presented for finding order quantities that result in a costs-optimal trade-off between shortages and outdating, by penalizing the occurrence of such events through the cost structure. They show that by a combination of optimization and simulation, the outdating of blood platelet pools (BPPs) can be reduced from 15% to 1% or even less fulfilling the same demand. In this section, we reflect on the role of simulation and briefly summarize the MDP model by defining the states, actions and transitions.

### 3.1 MDP Model for Perishable Inventory Management

We are interested in an optimal stock-age-dependent ordering policy for BPPs. The demand is uncertain with a mean that depends on the day of the week. Moreover the demand continues during the weekend, whereas the production during weekends is not possible. The maximum shelf life of BPPs is in The Netherlands $m = 5$ days counted from the moment the products are added to stock at the end of the day of production. Production order decisions are made daily except during weekends. Assuming that the problem is stationary across weeks, but not during a week, we thus deal with a periodic MDP with periodicity 7 days.

#### 3.1.1 States

Since production takes place during weekdays only, and demand is weekday dependent, the day of the week, $d \in \{1,2,\ldots,7\}$ for Monday to Sunday, has to be included in the state description. Further we need to keep track of the ageing of the products. This means that for recording outdating of BPPs, the state of the MDP contains information on the number of products in stock of each age category. The detailed information about the number of BPPs in stock of each age category is an $m$-dimensional vector, $\mathbf{x} \in \mathbb{N}^m$. Element $x_r$ is the number of BPPs in stock with a residual shelf life of $r$ days at the start of a day. A complete state description thus is the pair $(d,\mathbf{x})$.

#### 3.1.2 Decision and Decision Epoch

Every morning, except during weekends, a decision is taken about the number of BPPs ($a$) to produce during the day. In the MDP model, it is assumed that during the day the production volume is not revised, and at the end of the day the $a$ ordered BPPs are available to meet demand in the next $m$ days.

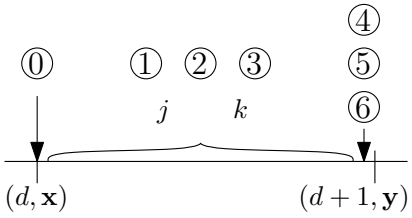
#### 3.1.3 Transitions

In the transition from day $d$ to the next day $d+1$ (where from now on $d+1=1$ if $d=7$), a sequence of events takes place as depicted in Figure 1. After the inspection of the stock state $\mathbf{x}$, the production volume $a$ is set early in the morning. Next the demand for BPPs is met. In fact, demand is split into two categories with different preferences regarding the age of the products to issuing policies: one category of demand is met by using the youngest products from stock (i.e. LIFO issuance), as this demand relates to patients that should be treated preferably with 'young' products with a residual shelf life of at least $r$ days. The other category gets issued the oldest products available (i.e. FIFO issuance) with a residual shelf life of

at least 1 day. The issuing policy $I$ is not part of the optimization problem as it is fixed for each demand category. When products are issued of an undesirable age, a mismatch is registered.

To summarize, the following events take place during a day:

1. BPPs are issued to meet the LIFO demand for $j$ 'young' BPPs and the FIFO demand for $k$ BPPs,
2. mismatches occur when young-demand is met by BPPs with a residual shelflife of less than $r$ days,
3. shortages occur when not all demand can be met from stock,
4. at the end of the day all BPPs age 1 day,
5. BPPs that become outdated are removed from stock,
6. the production is released and added to stock.




0 = State $(d, \mathbf{x})$ is inspected and production volume $a$ is set,
1 = Meet demand for $j$ 'young' and $k$ BPPs of any age,
2 = Mismatches may happen,
3 = Shortages may arise,
4 = BPPs in stock have aged one day,
5 = Some BPPs may outdate and are removed from stock,
6 = Production is released and added to final stock $\mathbf{y}$.

Figure 1: The chronological order of events in the MDP model.

### 3.2 Using Simulation to Solve the MDP

#### 3.2.1 Setting a Grid for the State Space

In order to solve the MDP a discrete and finite state space should be defined. In this case, the state space is already discrete. However, it is unbounded as we did not set a maximum number of products that can be kept in stock, nor have we specified an upper bound of the demand. By simulation of a simple sub optimal policy an upper bound of the state space may be observed. we argue that the maximum inventory level observed in the simulation may also be a reasonable upper bound for the MDP, if that simple policy results in almost no shortages but outdating on almost every weekday. Thus simulation helps in bounding and cutting the state space.

For example, if this maximum level is 40, we may assume that $\sum_{r=1}^{m} x_r \leq 40$. Thus we can set a grid for the state space and eliminate states that violate the artificial stock capacity constraint. In addition we may impose an artificial production capacity to limit the possible values $x_r$ for every $r \in \{1,2,\ldots,m\}$ may take. Next, we can determine the number of feasible grid points. Based on experience in solving MDPs of this type, we know that if this number of states is in the order of a few million or less, we are able to compute an optimal policy in a reasonable time. If this number is larger, we should choose a coarser grid by modeling the ordering and the demand in batches rather than in terms of individual products. Simulation is a fast procedure that generates insights for setting a suitable grid for the state space, but requires some intuition or trial and error for setting a reasonable policy that provides the bounds on and cuts in the state space. After solving the MDP, we may simulate the optimal policy to check whether the bounds on the state space are justified (i.e. non-binding).

#### 3.2.2 Discovering Structure of an Optimal Policy

Without going into the details of the value of problem parameters (which can be found in (Haijema 2008), Chapter 3), we present how an optimal order policy typically may look like. Table 1 presents the optimal actions as derived by value iteration for a selection of the states. All mentioned states represent the situation of having 14 products in stock on Wednesday morning. The optimal policy prescribes to produce 3, 4, 5, 6,

Table 1: Optimal production volumes on Wednesday under $I = (\text{LIFO}, \text{FIFO})$ for a selection of stock states with in total $x = 14$ BPPs in stock. ($x_2 = x_3 = 0$ due to the production stop in the weekends).

| Stock state **x** | | | | | Opt. prod. | Stock state **x** | | | | | Opt. prod. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | volume $a$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | volume $a$ |
| 0 | 0 | 0 | 3 | 11 | 4 | 3 | 0 | 0 | 5 | 6 | 5 |
| 1 | 0 | 0 | 5 | 8 | 3 | 4 | 0 | 0 | 1 | 9 | 6 |
| 2 | 0 | 0 | 4 | 8 | 4 | 4 | 0 | 0 | 4 | 6 | 6 |
| 2 | 0 | 0 | 5 | 7 | 3 | 5 | 0 | 0 | 1 | 8 | 7 |
| 3 | 0 | 0 | 1 | 10 | 5 | 6 | 0 | 0 | 2 | 6 | 7 |

or 7 products, depending on the age distribution of the 14 products, i.e. how the 14 products are distributed over the $m = 5$ age categories. The optimal policy thus seems to be rather complex. Approximating it by a simple rule such as an order-up-to $S$ rule or a fixed order quantity seems to be inappropriate.

The optimal solution consists of (*state,action*)-pairs like in Table 1 but it does not provide information on how likely it is that a state will be visited when the optimal policy is followed. Some states occur much more frequently than others. Hence if we want to approximate the complex stock-age dependent policy by a simpler stock-level dependent rule, we need to focus on the states that will occur most frequently.

Simulation appears to be a fast technique to identify the most frequently visited states. Simulation of an optimal policy for say 100,000 weeks provides a frequency of each stock state value. Frequency tables can be made for each weekday indicating how often each total stock level is observed and how often a specific number of products was ordered. In Table 2 the order quantity is translated into an order-up-to level $S$, as order-up-to $S$ rules are commonly used for non-perishables and therefore seem to be a good starting point for our analysis. In the first column of each subtable, one reads order-up-to levels. In the last column one reads how often a (*state, action*)-pair was visited that corresponds to the order-up-to level of the first column. For example on Wednesdays, order-up-to level 18 is optimal on 74,498 of the 100,000 simulated Wednesdays, hence about 75% of the time. On Mondays, however, the most frequently observed order-up-to level is 16, which appeared to be optimal on 54.5% of the 100,000 simulated Mondays. An order-up-to rule with a fixed weekday dependent order-to-level $S$, seems to fit reasonably well on Tuesdays, Thursdays and Fridays, but appears to be a bad fit on Mondays.

### 3.3 Simulation to Discover a New Class of Order Policies

The optimization-simulation approach sketched above, is used in (Haijema 2011) to discover a new class of order policies for perishables. In the frequency tables of some days a lower bound $q$ and an upper bound $Q$ on the order quantity is observed. By adding these bounds to the well known periodic review (s,S) policy, a new class of order policies is constructed that is named the (periodic review) $(s,S,q,Q)$ policy. The order quantity set by the $(s,S,q,Q)$ policy is 0, if the total stock level $x$ is above $s$, otherwise it is $\max\{q, \min\{S-x, Q\}\}$. The order threshold $s$ has no effect when it is set to $\infty$. The class contains three well known periodic review policies: the order-up-to $S$ policy, the fixed order quantity, and the $(s,S)$ policy. Besides inducing this new class of ordering policies that better resemble the optimal ordering policy of perishables, the simulation tables provide nearly optimal parameter values for $s$, $S$, $q$ and $Q$.

For example, Figure 2 shows how different rules may approximate the optimal order policy on Mondays. The sum of the frequency in the grey cells indicate how well a rule approximates the optimal policy. The best fitting parameters on Mondays appear to be $s = \infty$, $S = 16$, $q = 5$, and $Q = 7$. This observation would not have been made without simulating the optimal policy.

Table 2: Simulation frequency tables of scaled MDP strategy under $I = (\text{LIFO}, \text{FIFO})$. $S_d = 0$ indicates zero production.

(a) (State, action)-frequency tables for 100,000 simulated Mondays.

| Stock $x$ | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | Freq($S_1$) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Up-to $S_1$ | | | | | | | | | | | | | | | |
| 22 | | | | | | | | | | | | | | 1 | 1 |
| 21 | | | | | | | | | | | | | 10 | | 10 |
| 20 | | | | | | | | | | | 2 | 43 | | | 45 |
| 19 | | | | | | | | | 5 | 12 | 314 | | | | 331 |
| 18 | | | | | | | | 26 | 61 | 1366 | | | | | 1453 |
| 17 | | | | | | | 76 | 213 | 4383 | | | | | | 4672 |
| 16 | | | | | | 24465 | 19514 | 10558 | | | | | | | 54537 |
| 15 | | | | | 21002 | | | | | | | | | | 21002 |
| 14 | | | | 12498 | | | | | | | | | | | 12498 |
| 13 | | | 4551 | | | | | | | | | | | | 4551 |
| 12 | | 824 | | | | | | | | | | | | | 824 |
| 11 | 76 | | | | | | | | | | | | | | 76 |
| 10 | | | | | | | | | | | | | | | 0 |
| : | | | | | | | | | | | | | | | : |
| 0 | | | | | | | | | | | | | | | 0 |
| Freq($x$) | 76 | 824 | 4551 | 12498 | 21002 | 24465 | 19590 | 10797 | 4449 | 1378 | 316 | 43 | 10 | 1 | 100000 |

(b) (State, action)-frequency tables for 100,000 simulated Tuesdays.

| Stock $x$ | 0…5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | Freq($S_2$) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Up-to $S_2$ | | | | | | | | | | | | | | | |
| 26 | | | | | | | | | | | | | | 1 | 1 |
| 25 | | | | | | | | | | | | | 2 | | 2 |
| 24 | | | | | | | | | | | | 9 | | | 9 |
| 23 | | | | | | | | | | 1 | 28 | | | | 29 |
| 22 | | | | | | | | 2 | 2 | 170 | 1 | | | | 175 |
| 21 | | | | | | 3 | 6 | 15 | 563 | 21 | 5 | | | | 613 |
| 20 | | | | | 9 | 31 | 75 | 1479 | 209 | 35 | | | | | 1838 |
| 19 | | | | 31 | 127 | 213 | 2968 | 913 | 326 | | | | | | 4578 |
| 18 | | | 17963 | 20849 | 25014 | 19853 | 7626 | 1450 | | | | | | | 92755 |
| 17 | | | | | | | | | | | | | | | 0 |
| : | | | | | | | | | | | | | | | : |
| 0 | | | | | | | | | | | | | | | 0 |
| Freq($x$) | 0 | 0 | 17963 | 20880 | 25150 | 20100 | 10675 | 3859 | 1100 | 227 | 34 | 9 | 2 | 1 | 100000 |

(c) (State, action)-frequency tables for 100,000 simulated Wednesdays.

| Stock $x$ | 0…8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | Freq($S_3$) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Up-to $S_3$ | | | | | | | | | | | | | | | |
| 26 | | | | | | | | | | | | | | 1 | 1 |
| 25 | | | | | | | | | | | 4 | 2 | | | 6 |
| 24 | | | | | | | | | | 28 | 9 | 1 | 1 | | 39 |
| 23 | | | | | | | | | 91 | 62 | 21 | 4 | | | 178 |
| 22 | | | | | | | | 458 | 224 | 79 | 12 | 1 | | | 774 |
| 21 | | | | | | | 1301 | 753 | 243 | 34 | 1 | | | | 2332 |
| 20 | | | | | | 3129 | 1880 | 697 | 84 | 17 | 6 | | | | 5813 |
| 19 | | | | | 4745 | 3740 | 2284 | 679 | 140 | 13 | | | | | 11601 |
| 18 | | | | 12888 | 25368 | 23761 | 11559 | 790 | 112 | 20 | | | | | 74498 |
| 17 | | | | | | | 695 | 3199 | 598 | | | | | | 4492 |
| 16 | | | | | | | | 85 | | | | | | | 85 |
| 15 | | | | | | | | | | | | | | | 0 |
| : | | | | | | | | | | | | | | | : |
| 1 | | | | | | | | | | | | | | | 0 |
| 0 | | | | | | | | | 98 | 81 | 2 | | | | 181 |
| Freq($x$) | 0 | 0 | 0 | 12888 | 30113 | 30630 | 17719 | 6661 | 1590 | 334 | 55 | 8 | 1 | 1 | 100000 |

(d) (State, action)-frequency tables for 100,000 simulated Thursdays

| Stock $x$ | 0…3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | Freq($S_4$) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Up-to $S_4$ | | | | | | | | | | | | | | | |
| 17 | | | | | | | | 399 | 46 | | | | | | 445 |
| 16 | | | | | 4126 | 15584 | 26550 | 25562 | 1256 | 108 | 24 | 3 | | | 73213 |
| 15 | | | | 155 | | | | 314 | 15703 | 7305 | 2269 | 517 | | | 26263 |
| 14 | | | 3 | | | | | | | | | | | | 3 |
| 13 | | | | | | | | | | | | | | | 0 |
| : | | | | | | | | | | | | | | | : |
| 0 | | | | | | | | | | | | | 69 | 7 | 76 |
| Freq($x$) | 0 | 0 | 3 | 155 | 4126 | 15584 | 26550 | 26275 | 17005 | 7413 | 2293 | 520 | 69 | 7 | 100000 |

(e) (State, action)-frequency tables for 100,000 simulated Fridays.

| Stock $x$ | 0…3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | Freq($S_5$) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Up-to $S_5$ | | | | | | | | | | | | | | | |
| 23 | | | | | | | | | | | 4 | 2 | 2 | 1 | 9 |
| 22 | | | | | | | | | | 56 | 46 | 34 | 8 | 4 | 148 |
| 21 | | | | | | | | 44 | 345 | 831 | 634 | 944 | 88 | | 2886 |
| 20 | | | | | | 3591 | 17946 | 30836 | 27269 | 13380 | 3935 | | | | 96957 |
| 19 | | | | | | | | | | | | | | | 0 |
| : | | | | | | | | | | | | | | | : |
| 0 | | | | | | | | | | | | | | | 0 |
| Freq($x$) | 0 | 0 | 0 | 0 | 0 | 3591 | 17946 | 30880 | 27614 | 14267 | 4619 | 980 | 98 | 5 | 100000 |

| Stock $x$ | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | Freq($S_d$) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Up-to $S_1$ | | | | | | | | | | | | | | | |
| 22 | | | | | | | | | | | | | | 1 | 1 |
| 21 | | | | | | | | | | | | | 10 | | 10 |
| 20 | | | | | | | | | | | 2 | 43 | | | 45 |
| 19 | | | | | | | | | 5 | 12 | 314 | | | | 331 |
| 18 | | | | | | | | 26 | 61 | 1366 | | | | | 1453 |
| 17 | | | | | | | 76 | 213 | 4383 | | | | | | 4672 |
| 16 | | | | | | 24465 | 19514 | 10558 | | | | | | | 54537 |
| 15 | | | | | 21002 | | | | | | | | | | 21002 |
| 14 | | | | 12498 | | | | | | | | | | | 12498 |
| 13 | | | 4551 | | | | | | | | | | | | 4551 |
| 12 | | 824 | | | | | | | | | | | | | 824 |
| 11 | 76 | | | | | | | | | | | | | | 76 |
| 10 | | | | | | | | | | | | | | | 0 |
| : | | | | | | | | | | | | | | | : |
| 0 | | | | | | | | | | | | | | | 0 |
| Freq($x$) | 76 | 824 | 4551 | 12498 | 21002 | 24465 | 19590 | 10797 | 4449 | 1378 | 316 | 43 | 10 | 1 | 100000 |

(a) Reading an order-up-to $S$ rule with a fixed order-up-to level.

| Stock $x$ | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | Freq($S_d$) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Up-to $S_1$ | | | | | | | | | | | | | | | |
| 22 | | | | | | | | | | | | | | 1 | 1 |
| 21 | | | | | | | | | | | | | 10 | | 10 |
| 20 | | | | | | | | | | | 2 | 43 | | | 45 |
| 19 | | | | | | | | | 5 | 12 | 314 | | | | 331 |
| 18 | | | | | | | | 26 | 61 | 1366 | | | | | 1453 |
| 17 | | | | | | | 76 | 213 | 4383 | | | | | | 4672 |
| 16 | | | | | | 24465 | 19514 | 10558 | | | | | | | 54537 |
| 15 | | | | | 21002 | | | | | | | | | | 21002 |
| 14 | | | | 12498 | | | | | | | | | | | 12498 |
| 13 | | | 4551 | | | | | | | | | | | | 4551 |
| 12 | | 824 | | | | | | | | | | | | | 824 |
| 11 | 76 | | | | | | | | | | | | | | 76 |
| 10 | | | | | | | | | | | | | | | 0 |
| : | | | | | | | | | | | | | | | : |
| 0 | | | | | | | | | | | | | | | 0 |
| Freq($x$) | 76 | 824 | 4551 | 12498 | 21002 | 24465 | 19590 | 10797 | 4449 | 1378 | 316 | 43 | 10 | 1 | 100000 |



(b) Reading a bounded order-up-to $S$ rule with bounded order quantities.

| Stock $x$ | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | Freq($S_d$) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Up-to $S_1$ | | | | | | | | | | | | | | | |
| 22 | | | | | | | | | | | | | | 1 | 1 |
| 21 | | | | | | | | | | | | | 10 | | 10 |
| 20 | | | | | | | | | | | 2 | 43 | | | 45 |
| 19 | | | | | | | | | 5 | 12 | 314 | | | | 331 |
| 18 | | | | | | | | 26 | 61 | 1366 | | | | | 1453 |
| 17 | | | | | | | 76 | 213 | 4383 | | | | | | 4672 |
| 16 | | | | | | 24465 | 19514 | 10558 | | | | | | | 54537 |
| 15 | | | | | 21002 | | | | | | | | | | 21002 |
| 14 | | | | 12498 | | | | | | | | | | | 12498 |
| 13 | | | 4551 | | | | | | | | | | | | 4551 |
| 12 | | 824 | | | | | | | | | | | | | 824 |
| 11 | 76 | | | | | | | | | | | | | | 76 |
| 10 | | | | | | | | | | | | | | | 0 |
| : | | | | | | | | | | | | | | | : |
| 0 | | | | | | | | | | | | | | | 0 |
| Freq($x$) | 76 | 824 | 4551 | 12498 | 21002 | 24465 | 19590 | 10797 | 4449 | 1378 | 316 | 43 | 10 | 1 | 100000 |

(c) Reading a fixed order quantity.

Figure 2: Reading simple rules from $(state, action)$-frequency tables of optimal production policy (for Monday) under (LIFO, FIFO) issuing.

## 4 CASE 2: SETTING FISHING QUOTA

### 4.1 Markov Decision Problem (MDP)

To maintain a sustainable fish stock, a European authority regulates the fishing behavior of the European fishery sector by setting fishing quota. The determination of fishing quota can be studied by an infinite horizon dynamic program. In van Dijk, Hendrix, Haijema, Groeneveld, and van Ierland 2012 a bio-economic MDP model is presented that contains fish stock dynamics as well as capital development in fishing equipment. In the dynamic programming context, the state vector $\mathbf{x}$ contains the fish and capital stock $(S,K)$ and the decision is the quota $Q(\mathbf{x})$ to be set for a specific species. The authors of (van Dijk, Haijema, Hendrix, Groeneveld, and van Ierland 2012) study the implications of introducing a constraint on the adjustment of European quota. Specifically, the authority can adjust the quota in period $t$ within a range of 15% lower or higher than the previously set quota $Q_{t-1}$. The presumption made in practice of this constraint is that it would lead to a more stable fish stock and to more stability for fishermen who have to decide about capital investments. The MDP model is slightly complicated by the quota adjustment constraint, as $Q_{t-1}$ should be added to the state space. We discuss the model and the implications of solving it in the sequel.

For the details of the model we refer to (van Dijk, Hendrix, Haijema, Groeneveld, and van Ierland 2012) and the references therein. The new aspect of the used model with respect to literature is that the authors use a so-called bi-level model. On the lower level, fishermen react on the fish stock $S$ and set quota $Q$ and decide on the harvest $H$ and investment $I$ given the capital $K$ they have at that moment, i.e. in a non-dynamic way. The authority, keeps the long term welfare into account on the first level and sets quota $Q$ given the behavior of fishermen on the second level. The fish stock dynamics is described by the stochastic process

$$S_{t+1} = S_t + \xi\,(rS_t\left(1-\frac{S_t}{m}\right)) - H_t, \tag{2}$$

where $m$ is the so-called carrying capacity, $r$ the intrinsic growth rate, and $\xi$ a lognormal distributed variable with mean 1. The capital stock follows the behavior of the fishermen that react in a deterministic way on the state values and the quota

$$K_{t+1} = K_t(1-d) + I_t, \tag{3}$$

where $d$ is a depreciation rate.

### 4.2 Analyzing the MDP Model by Simulation

Note that the state of the fish stock as well as the capital position of the fishermen is a continuous state variable. In order to solve the MDP numerically, a grid over the state space is constructed. Therefore, we analyze the problem and make use of simulation to explore the boundaries of the system. An important detail in the harvest intensity is the use of the so-called Spence function, see Spence 1973. The Spence function describes that it takes more effort to harvest a given amount of fish, if the fish stock $S$ is low. Formulated the other way around, given the capacity of the fishing fleet, the amount of fish harvested is less when the fish stock is low. There is even a fish stock level $\hat{s}$, under which harvest is not economically viable; the cost of sailing is higher than the sales. Simulations of the system for any quota setting rule show that fish dynamics combined with this Spence harvest behavior leads to an extremely stable system.

The behavior of the system is depicted in Figure 3. From these simulations one can capture that levels higher than the carrying capacity $m$ always lead to a reduction of stock and levels lower than level $\hat{s}$, always lead to an increase in stock, as fishermen do not harvest no matter what quota are set.

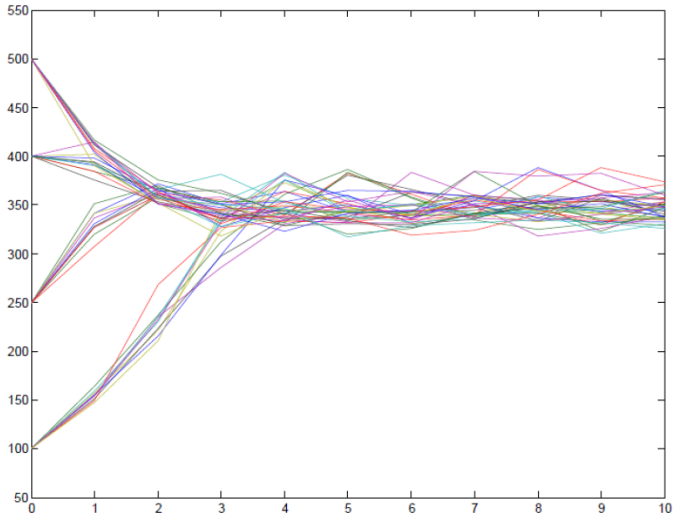



Figure 3: Sample paths of fish stock for 4 different starting values of fish stock.

### 4.2.1 Constructing a Grid for the State Space

So, $\hat{s}$ is a lower bound of the fish stock $S_t$, if the initial stock $S_0 > \hat{s}$. At the other side, if the initial stock is higher than the carrying capacity, $S_0 > m$, then the stock can only go down from that level. Given an initial stock $S_0$, we have that in the system

$$S_t \in [\min\{S_0, \hat{s}\}, \max\{S_0, m\}]. \tag{4}$$

This means that the interesting range for harvest $H$ and quota $Q$ is $H, Q \in [0, \max\{S_0, m\} - \hat{s}]$.

These bounds provide a range of appropriate values for the capital stock $K$ and investment $I$. Due to investment cost and depreciation, the level of capital should not exceed what is required to catch the desired level, as specified by the Spence harvest function (Spence 1973);

$$K, I \in [0, \max\{\frac{1}{q}\ln(\frac{m}{\hat{s}}), K_0\}], \tag{5}$$

where $q$ is a so-called catchability coefficient.

The observed boundaries of the system are of utmost importance for bounding the system and designing the grid on which the value function iteration procedure is run.

### 4.2.2 Solution Process

First of all, the optimization of $Q$ in the Bellman equation does not require a grid search approach. One can run a one-dimensional nonlinear optimization routine for each grid point in the $(S,K)$ space. With respect to the ranges (4) and (5), choosing a wider range to be "more certain" leads to slow convergence of the value iteration, because one includes in fact states that have a low probability of occurrence. When evaluating the 15% rule, also the state variable $Q_{t-1}$ has to be included in the state space using a grid with the same range as that of decision variable $Q$.

The last challenge is that one is dealing with a continuous random variate that in principle has an unbounded support. The usual approach is to discretize the space of possible outcomes of the stochastic variable. A sharp way to do so is by using the quantiles of the lognormal distribution. Practically this works by using an equidistant grid over the probability range $[0,1]$ with a step $p_\xi$ and generating a discrete outcome space $\{\theta_1, \theta_2, \ldots, \theta_n\} = \{G^{-1}(p_\xi), G^{-1}(2p_\xi), G^{-1}(3p_\xi), \ldots, G^{-1}(1-p_\xi)\}$. The consequence of

this operation is that the outcome space is truncated by the $p_\xi$-quantiles and every outcome has the same probability of occurrence. Actually, the expected contributions from period $t+1$ onwards in a Bellman equation is now approximated by a sum on possible outcomes:

$$p_\xi \sum_{i=1}^{n} V_{t+1}\left(S+\theta_i(rS\left(1-\frac{S}{m}\right))-H(Q,S,K),(1-d)K+I(Q,S,K)\right). \tag{6}$$

Notice, that one requires interpolation in the grid values of the multi-dimensional state space in order to valuate the Bellman equation. A result of the determination of the table of optimal quota from the study of van Dijk, Hendrix, Haijema, Groeneveld, and van Ierland 2012 is given in Figure 4.

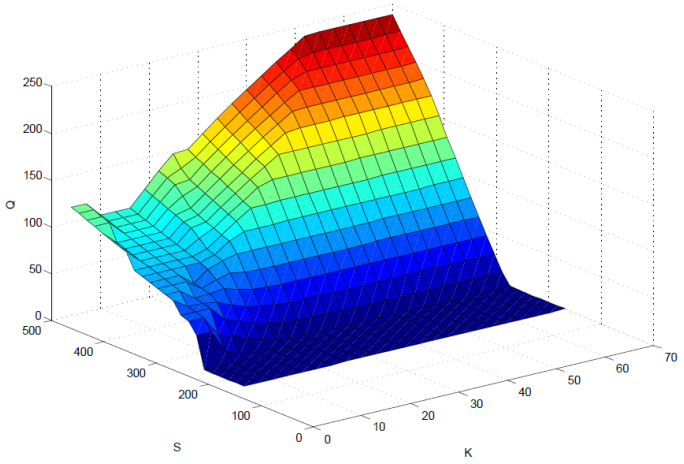



Figure 4: Optimal quota after value iteration on $S,K$ state space.

The optimal quota for the case that the authority introduces the 15% rule cannot be blindly compared, as we are dealing with a higher dimensional state space. Here simulation comes at help and allows us to compare the long term average when the system begins at several starting values of the system. Figure 5 shows how the quota evolves over time when starting with a relatively large fish stock and a low initial quota. Due to the quota adjustment constraint the solid line overlaps the dashed +15% line in the first 5 years, but thereafter the quota is quite stable.

One can observe that after reaching a stationary state, in fact the included restriction is not binding. This means, that only after a shock in the system, inclusion of such a rule may be binding. van Dijk, Haijema, Hendrix, Groeneveld, and van Ierland 2012 study the development of the investment decision, harvest and fish stock for several assumptions on fishermen's behavior after a shock takes place. The observed characteristic of the model is a large stability due to the growth function of the fish stock dynamics and the Spence harvest function. Only if due to shocks the fish stock deviates from the steady state value, the inclusion of a restricting rule provides more stability to the fishery sector.

## 5 DISCUSSION AND CONCLUSION

We have discussed how simulation is an aid to solve dynamic programs, in particular, Markov decision problems (MDP). This has been illustrated by two case studies. Three ways are shown how simulation may contribute solving MDPs. First, simulation discovers the natural boundary of the system. This provides a tool to construct a grid for the state space, which is usually made discrete and finite as MDPs are usually solved numerically. Second, simulation helps analyzing the properties of a system that operates under a computed optimal policy. That way, simulation is a tool to validate assumptions made about the MDP model. Third, simulation helps discovering approximate policies that are close to optimal but that are easier

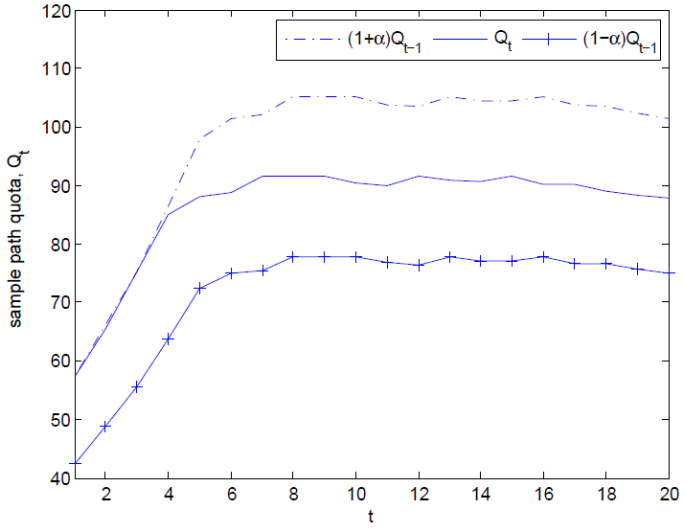



Figure 5: Quota dynamics in 20 simulated years where changes in quota are limited to $\alpha = 15\%$.

to understand. For the case of the inventory management of perishable the simulation based analysis led to the discovery of a new class of ordering policies, as well as to a procedure to determine nearly optimal parameter values.

In this paper, we limited ourselves to large sequential optimization problems that in principle can be solved to optimality by classical techniques like value iteration (maybe after scaling or downsizing the problem). In practice, one may find other problems that cannot be solved by these techniques as the state space is too large, e.g. with hundreds or thousands of dimensions. In the literature one finds that in those cases simulation can also be very helpful. Recently developed techniques like approximate dynamic programming (see (Powell 2007)) make use of simulation, optimization and learning techniques to approximate value functions from which one attempts to deduce nearly optimal actions (with no guarantee of optimality though). Despite the effort put in this area, there is no golden rule on how to solve such large optimization problems. Solving large MDPs requires a problem specific approach that exploits the structure present in the problem. Simulation is a powerful tool to search for this problem structure and may lead to good results in combination with existing optimization techniques. The way simulation is used in solving the MDP for the two cases presented in this paper may stimulate researchers in solving others MDPs by a combination of simulation and optimization.

## ACKNOWLEDGMENTS

This work has been funded by TI Food and Nutrition (TIFN project RE002) and grants from the Spanish Ministry of Science and Innovation (TIN2008-01117) and Junta de Andalucía (P11-TIC-7176), in part financed by the European Regional Development Fund (ERDF).

## REFERENCES

Bellman, R. 1957. *Dynamic Programming*. Princeton University Press.

Haijema, R. 2008, December. *Solving Large Structured markov decision problems for perishable inventory management and traffic control*. Ph. D. thesis, University of Amsterdam - Tinbergen Institute - Amsterdam School of Economics.

Haijema, R. 2011. "A new class of stock level dependent ordering policies for perishables with a short maximum shelf life". *International Journal of Production Economics*. doi:10.1016/j.ijpe.2011.05.021.

Haijema, R., J. van der Wal, and N. M. van Dijk. 2007, March. "Blood platelet production: Optimization by dynamic programming and simulation". *Computers and Operations Research* 34 (3): 760–779. doi:10.1016/j.cor.2005.03.023.

Haijema, R., N. M. van Dijk, J. van der Wal, and C. Smit Sibinga. 2009. "Blood Platelet Production with Breaks: Optimization by SDP and Simulation". *International Journal of Production Economics* 121:467–473. doi:10.1016/j.ijpe.2006.11026.

Howard, R. A. 1960. *Dynamic Programming and Markov Processes*. MIT Press, Cambridge University.

Karaesmen, I., A. Scheller-Wolf, and B. Deniz. 2011. *Planning Production and Inventories in the Extended Enterprise*, Volume 151 of *International Series in Operations Research & Management Science*, Chapter 15. Managing Perishable and Aging Inventories: Review and Future Research Directions, 393–436. Springer.

Nahmias, S. 1982. "Perishable Inventory Theory: A Review". *Operations Research* 30:680–708.

Powell, W. B. 2007. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley Series in Probability and Statistics. Wiley.

Puterman, M. L. 1994. *Markov decision processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Mathematical Statistics.

Spence, M. 1973. "Blue Whales and Applied Control Theory". Technical Report 108, Institute for Mathematical Studies, Stanford University.

van Dijk, D., R. Haijema, E. M. T. Hendrix, R. Groeneveld, and E. van Ierland. 2012. "A bio-economic evaluation of the 15%-rule in EU fish quota: obtaining a sustainable fish stock and planning capital investments". working paper wwp-6, Wageningen School of Social Sciences.

van Dijk, D., E. M. T. Hendrix, R. Haijema, R. Groeneveld, and E. van Ierland. 2012. "A tutorial on bio-economic SDP modeling: an illustration of fisheries policies". working paper wwp-4, Wageningen School of Social Sciences.

## AUTHOR BIOGRAPHIES

**RENE HAIJEMA** is assistant professor at the Operations Research and Logistics group at Wageningen University, The Netherlands. He is also scientists at TI Food and Nutrition, Wageningen, The Netherlands. His main research interest is in simulation and stochastic optimization in the area of (food) logistics, especially in perishable inventory management. He received a MSc and PhD degree in Operations Research and Management from the University of Amsterdam, The Netherlands. During his PhD project he has solved large structured Markov Decision Problems by dynamic programming and simulation for inventory management and traffic control. His email address is Rene.Haijema@wur.nl.

**ELIGIUS M. T. HENDRIX** is a Ramón y Cajal researcher at the Computer Architecture department, Málaga University. He is also affiliated with Wageningen University and specifically interested in (global) optimization algorithms and solution processes for decision problems. He holds a MSc and BSc degree from Tilburg University and a PhD degree from Wageningen University. His email address is eligius.hendrix@wur.nl.

**DIANA VAN DIJK** is PhD candidate at the Environmental and Natural Resources group of Wageningen University. She holds an MSc degree in Environmental Economics from the same university. In 2012 she expects to graduate on a thesis devoted to the optimization of fishing quota adjustments. Her email address is Diana.VanDijk@wur.nl.

**JAN VAN DER WAL** is professor in Operations Research at the University of Amsterdam. He holds a PhD degree from Eindhoven University of Technology. His email address is j.vanderwal@uva.nl.