INTERNATIONAL JOURNAL OF ACADEMIC RESEARCH IN ACCOUNTING, FINANCE AND MANAGEMENT SCIENCES

www.hrmars.com
ISSN: 2225-8329

Dynamic Programming – A Profit Optimization Method

Delia Teselios, Mihaela Albici

To Link this Article: http://dx.doi.org/10.6007/IJARAFMS/v3-i1/10294 DOI:10.6007/IJARAFMS /v3-i1/10294

Received: *14* January *2013, **Revised:** 17* February 2013*, **Accepted:** 25* February 2013

Published Online: 19 March 2013

In-Text Citation: (Teselios & Albici, 2013)
To Cite this Article: Teselios, D., & Albici, M. (2013). Dynamic Programming – A Profit Optimization Method. *International Journal of Academic Research in Accounting Finance and Management Sciences*, *3*(1), 351–358.

Vol. 3, No. 1, 2013, Pg. 351 - 358

http://hrmars.com/index.php/pages/detail/IJARAFMS **JOURNAL HOMEPAGE**

INTERNATIONAL JOURNAL OF ACADEMIC RESEARCH IN ACCOUNTING, FINANCE AND MANAGEMENT SCIENCES
www.hrmars.com
ISSN: 2225-8329

Dynamic Programming – A Profit Optimization Method

Delia Teselios, Mihaela Albici
Faculty of Management Marketing in Economic Affairs, "Constantin Brancoveanu" University, 2A Calea Bascovului Street, Pitesti, Romania
Email:delia_teselios@yahoo.com, mturmacu@yahoo.com

Abstract
Dynamic programming techniques are often used in economy due to the recursive structure that many dynamic economic optimization problems have. These problems, usually having a complex form, are disintegrated into smaller sub-problems whose optimal solutions lead to the optimal solution of the original problem. In the economic study of this paper we use the backward method in which the final state of the system and the chosen policy determine the initial state of the system.
Keywords: Dynamic Programming, Optimal Policy, Transfer Function, Efficiency Function, Backward Method

Introduction

The term "dynamic programming" was used in the '50s by the American mathematician Richard Bellman, in solving problems arising in the study of sequential decision – making processes. Bellman chose the expression "dynamic programming" to emphasize the appearance of time variation of processes, in the classical physical sense (Dreyfus, 2002).

In the paper "The theory of Dynamic Programming" (Bellman, 1954) are some examples of processes that allow dynamic programming: planning of industrial production line, determining replacement policy of equipment in factors, scheduling patients at a medical clinic, determining long-term investment programs for universities, etc.

Used in mathematics, informatics, economy, this method consists in transforming a complex initial sequential problem into a sequence of simpler, smaller problems, whose solutions are optimal and lead to the optimal solution of the original problem.

The method is based on the principle of optimality (Bellman, 1954): "An optimal policy has the property that whatever the initial state and decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decisions", where, by policy, we understand "a sequence of decisions […] which is most advantageous according to some preassigned criterion".

In other words "any optimal policy can only be formed from optimal sub-policies (Kaufmann, 1967).

Aspects of Backward Dynamic Programming

We consider a sequential decision process whose evolution over time can be controlled by the actions of a decision making factor and describe the system's state by vector $(s_t)_{t\in R}$ called state vector (Trandafir, 2004).

The notations that we operate with in this paper are those used in (Popescu et al., 1999). Thus,

S_t represents the set of possible states at time t;

X_t represents the set of admissible solutions, made up of solutions (policies, decisions) x_t taken by the decision-maker at time t;

$$u_{i+1} = u_{i+1}(s_i, x_{i+1}) \quad (1)$$

Where: $i = \overline{1,n}$ is called efficiency function and measures the quality of decision x_{i+1};

$f = f(u_1(s_0, x_1), \ldots, u_n(s_{n-1}, x_n))$ is the overall efficiency function;

$\tau_{i+1}(s_i, x_{i+1}) = s_{i+1}$ is called transfer function from state i in state $i + 1$.

Using recursive writing we obtained:

$$s_{i+1} = \tau_{i+1}(\tau_i(\ldots\tau_1(s_0, x_1)\ldots), x_{i+1}) \quad (2)$$

A number of important features of dynamic programming are presented in (Chinneck, 2012).

Thus:

- The problem to be solved can be decomposed into subproblems (which are solved independently), each corresponding to a stage representing the moments when decisions must be taken. Depending on the directions of the scroll, we have forward dynamic programming or backwards dynamic programming;

- Each stage has a certain number of states, representing the information needed in solving subproblems;

- After making a decision, the system state changes as the relationship

$$\tau_{i+1}(s_i, x_{i+1}) = s_{i+1} \quad (3)$$

- Given the current state, the optimal decision for the remaining stages is independent of decisions made in previous states. This statement represents the fundamental dynamic programming principle of optimality;

- There is a recursive relationship between the value of decision at a stage and the value of the optimum decisions at previous stages.

In the economic study of this paper we use the backwards method, in which the final state and the chosen policy determine the original state.

Definition: A sequential optimization problem is called backwards decomposable if there $F_i : R^2 \to R$ monotonic functions (increasing for an objective function "maxim" and decreasing for an objective function "minimum") in the second variable, so that:

$$f_{n-i+1}(u_i(s_{i-1},x_i),\ldots,u_n(s_{n-1},x_n)) = F_{n-i+1}(u_i(s_{i-1},x_i), f_{n-i}(u_{i+1}(s_i,x_{i+1}),\ldots,u_n(s_{n-1},x_n))) \quad (4)$$

with f_{n-i+1} efficiency function associated with decision process limited in phases $i, i+1, \ldots, n$ for $i = \overline{1, n-1}$ and $f_1(u_n) = u_n$ and $f_n = f$.

A policy consists of a sequence of decisions (Trandafir, 2004). The optimal policy s^* is the corresponding policy of the optimal decision x^* that makes the optimal objective function.

Economic Study

To increase profits, a fish farm manager decided to make an investment of 10.000 euro in the following objectives: a pond for sport fishing (noted by O_1), a pool for semi-intensive growth of trout (noted by O_2) and a small guesthouse (noted by O_3).

The question that arises in that of optimal allocation of the amount invested so that the obtained profit to be maximum, considering the data in Table 1.

Observation: The data from Table 1 have an illustrative role.

Table 1. The profit percentage, provided by investments

Objectives / Investment	Pond for sport fishing	Pool for semi-intensive growth	Small guesthouse
0	0	0	0
1	0,2	0,16	0,15
2	0,23	0,19	0,20
3	0,29	0,30	0,27
4	0,35	0,32	0,39
5	0,44	0,43	0,48
6	0,50	0,53	0,51
7	0,59	0,57	0,55
8	0,73	0,78	0,80
9	0,90	0,84	0,82
10	0,95	1,10	0,9

Source: Made by the author

Step 1: It is written the mathematical model of the problem.

We note:

x_i = Investment provided for objective O_i

$z_i(x_i)$ = Profit of investment x_i

s_i = Investment in objectives *i*

Objective function:

$$\max f = z_1(x_1) + z_2(x_2) + z_3(x_3) \quad (5)$$

The constraint is: $x_1 + x_2 + x_3 = 10$ (6)

Non-negativity conditions:

$x_i \geq 0,\ i = \overline{1,10}$ with $x_i =$ integer numbers.

Step 2: It is determined the set of possible states at time *i*, transfer functions and the set of feasible solutions.

$$s_0 = 0,\ s_0 \in S_0 = \{0\},\ \tau_1(s_0, x_1) = s_0 + x_1 \quad (7)$$

$$s_1 = s_0 + x_1 = x_1,\ s_1 \in S_1 = \{0,1,2,3,4,5,6,7,8,9,10\},\ \tau_2(s_1, x_2) = s_1 + x_2 \quad (8)$$

$$s_2 = s_1 + x_2 = x_1 + x_2,\ s_2 \in S_2 = \{0,1,2,3,4,5,6,7,8,9,10\},\ \tau_3(s_2, x_3) = s_2 + x_3 \quad (9)$$

$$s_3 = 10,\ s_0 \in S_3 = \{10\} \quad (10)$$

$$X_1(s_0) = \{0,1,2,3,4,5,6,7,8,9,10\} \quad (11)$$

$$X_2(s_1) = \{0 - s_1, 1 - s_1, \ldots, 10 - s_1\} \quad (12)$$

$$X_3(s_2) = \{10 - s_2\} \quad (13)$$

Step 3 : There are reestablished the efficiency functions and the objective functions of the subproblems obtained by dividing the original problem.

$$u_1(s_0, x_1) = z_1(x_1) \quad (14)$$

$$u_2(s_1, x_2) = z_2(x_2) \quad (15)$$

$$u_3(s_2, x_3) = z_3(x_3) \quad (16)$$

$$\tilde{f}_1 = z_1(x_1) + z_2(x_2) + z_3(x_3) \quad (17)$$

$$\tilde{f}_2 = z_2(x_2) + z_3(x_3) \quad (18)$$

$$\tilde{f}_3 = z_3(x_3) \quad (19)$$

Step 4: Since the problem is backwards decomposable with $F_i(m,n) = m + n$, $i = 2,3$, it is determined the optimal solutions of the component subproblems.

Let the recursive equations of backwards dynamic programming (Popescu et al., 1999) $g_{n-i+1}(s_{i-1}) = \max\{\tilde{f}_{n-i+1} / x_i \in X_i(s_{i-1})\}$ with the notations presented above, which we use in the following cases:

I. Case $n = 3$ and $i = 3$

$$g_1(s_2) = \max\{\tilde{f}_3 / x_3 \in X_3(s_2)\} = \max\{z_3(x_3)/x_3 \in X_3(s_2)\} = z_3(10 - s_2) \quad (20)$$

This value was obtained for $x_3^*(s_2) = 10 - s_2$

II. Case $n = 3$ and $i = 2$

$$g_2(s_1) = \max\{\tilde{f}_2 / x_2 \in X_2(s_1)\} = \max\{z_2(x_2) + z_3(x_3)/x_2 \in X_2(s_1)\} =$$

$$= \max\{z_2(x_2) + z_3(10 - s_2)/x_2 \in X_2(s_1)\}$$

$$= \max\{z_2(x_2) + z_3(10 - s_1 - x_2)/x_2 \in X_2(s_1)\} \quad (21)$$

Because s_1 covers the set {0,1,2,3,4,5,6,7,8,9,10} for $g_2(s_1)$ we obtain the values in Table 2.

Table 2. Values obtained for $g_2(s_1)$

F	G	H	J	K	L	M	N	O	P	Q	R	S	T
	x2	g2(0)	g2(1)	g2(2)	g2(3)	g2(4)	g2(5)	g2(6)	g2(7)	g2(8)	g2(9)	g2(10)	
	0	0.9											
	1	0.98	0.82										
	2	0.99	0.96	0.8									
	3	0.85	0.74	0.71	0.55								
	4	0.83	0.81	0.7	0.67	0.51							
	5	0.91	0.8	0.78	0.67	0.64	0.48						
	6	0.92	0.82	0.71	0.69	0.58	0.55	0.39					
	7	0.84	0.8	0.7	0.59	0.57	0.46	0.43	0.27				
	8	0.98	0.77	0.73	0.63	0.52	0.5	0.39	0.36	0.2			
	9	0.99	0.93	0.72	0.68	0.58	0.47	0.45	0.34	0.31	0.15		
	10	1.1	0.84	0.78	0.57	0.53	0.43	0.32	0.3	0.19	0.16	0	
maxim		**1.1**	**0.96**	**0.8**	**0.69**	**0.64**	**0.55**	**0.45**	**0.36**	**0.31**	**0.16**	**0**	

Source: Made by the author

The maximum values presented in Table 2 were calculated for:

$x_2^*(0)=10,\ x_2^*(1)=2,\ x_2^*(2)=2,\ x_2^*(3)=6,\ x_2^*(4)=5,\ x_2^*(5)=6,\ x_2^*(6)=9,\ x_2^*(7)=8,\ x_2^*(8)=9,$
$x_2^*(9)=10,\ x_2^*(10)=0$

III. Case $n=3$ and $i=1$

$$g_3(s_0)=\max\{\tilde{f}_3 / x_1 \in X_1(s_0)\}=\max\{z_1(x_1)+\tilde{f}_2 / x_1 \in X_1(s_0)\}=$$
$$=\max\{z_1(x_1)+g_2(s_1)/x_1 \in X_1(s_0)\}$$
$$=\max\{z_1(x_1)+g_2(s_0+x_1)/x_1 \in X_1(s_0)\}$$
$$=\max\{z_1(x_1)+g_2(x_1)/x_1 \in X_1(s_0)\}$$
(22)

For $g_3(s_0)$ are obtained the values in Table 3.

Table 3. Values obtained for $g_3(s_0)$

	x1	g3(s0)
36		
37	x1	g3(s0)
38	0	1.1
39	1	1.16
40	2	1.03
41	3	0.98
42	4	0.99
43	5	0.99
44	6	0.95
45	7	0.95
46	8	1.04
47	9	1.06
48	10	0.95
49		
50	maxim	1.16
51		
52		
53		

Source: Made by the author

The maximum of 1.16 was obtained for $x_1^* = x_1^*(s_0^*) = 1$, with $s_0^* = 0$

Analogically we obtain:

$$x_2^* = x_2^*(s_1^*) = x_2^*(s_0^* + x_1^*) = x_2^*(1) = 2$$

$$x_3^* = x_3^*(s_2^*) = x_3^*(s_1^* + x_2^*) = x_3^*(3) = 10 - s_2^* = 7$$

$$\Rightarrow \max f = z_1(x_1) + z_2(x_2) + z_3(x_3) = z_1(1) + z_2(2) + z_3(7) = 0,2 + 0,19 + 0,55 = 0,94$$

(23)

Conclusions

Thus, the optimal policy is (1,2,7). That is, the 10.000 euro will be invested as follows: 1.000 euro for the sport fishing pond, 2.000 euro for the semi-intensive growth pool and 7.000 euro for the small guesthouse.

References

Bellman, R. (1954). *The theory of dynamic programming*, The RAND Corporation, available online at http://www.rand.org/pubs/papers/2008/P550.pdf

Chinneck, W. J. (2012). *Practical optimization: A gentle introduction*, available online at http://www.sce.carleton.ca/faculty/chinneck/po.html

Dreyfus, S. (2002). *Richard Bellman on the Birth of Dynamic Programming*, Operations Research © 2002 INFORMS, Vol. 50, No. 1, pp. 48–51, from http://www.cas.mcmaster.ca/~se3c03/journal_papers /dy_birth.pdf

Kaufmann, A. (1967). *Metode si modele ale cercetarii operationale, vol. I,II*, Editura Stiintifica si Enciclopedica, Bucuresti

Trandafir, R. (2004). *Modele si algoritmi de optimizare*, Editura AGIR, Bucuresti, from http://civile.utcb.ro/mao.pdf