Simulation as an Aid to Dynamic Programming for Natural Resource Problems

Mark Eigenraam, Department of Natural Resources and Environment
Government of Victoria, e-mail: Mark.Eigenraam@nre.vic.gov.au
James E. Everett, Department of Information Management and Marketing,
The University of Western Australia, e-mail: jeverett@ecel.uwa.edu.au
Jason Barker, Department of Natural Resources and Environment
Government of Victoria, e-mail: Jason.Barker@nre.vic.gov.au

Abstract Dynamic programming (DP) can be used to analyse natural resources problems that have a temporal dimension. Model specification consists of two parts. Firstly, a transition system (specific to the natural resource) is applied at each time stage. Secondly, a dynamic programming algorithm identifies the optimal solution at each stage. The transition system is unique to each problem, while the dynamic programming technique is more generally applicable. Previous applications of DP have usually required expertise in program coding. Therefore each model required coding of the transition system and the DP algorithm. The application of DP can be simplified significantly if the two parts are decoupled. We use a graphical user interface (GUI) simulation package to decouple the transition system from the dynamic programming algorithm. The GUI allows each component part to be represented by a module. The transition module receives input state and decision vectors, and returns the appropriate output state vector. The DP module specifies dimensions of stages, states and decisions, and transformation of input state and decision vectors into the output state vector. The simulation advances one stage for each combination of input state and decision vectors. At the end of the simulation the DP module processes the transition output at each stage and reports the optimum decision for each stage. Using GUI to decouple the dynamic programming algorithm from the transition system allows the user to concentrate on the problem formulation, which can be connected to a standardised dynamic programming algorithm that suits natural resource problems. The method is illustrated in a case study of the control of wild oats. Interpolation methods to overcome the "curse of dimensionality" are considered for continuous state vectors, with decision vectors that may or may not be continuous.

1. INTRODUCTION

The Department of Natural Resources and Environment has the responsibility of advising the Victorian Government on a wide variety of natural resources policy issues, ranging from agriculture, fisheries, forestry to salinity, acidity and river health.

An important component of policy research is the study and modeling of dynamic natural resource systems. Typical features of such systems are:

- They are multi-*stage* (multiple time periods or temporal in nature), with each stage commonly representing a year or a season.
- They have multiple *state variables*, such as fish population, state of the soil, procreation rate, seed bank etc.
- The *output state(s)* at the end of each stage are determined by the input states at the beginning of the stage. The change in state from the beginning to the end of each stage is determined (calculated) by the transition equation.
- The transition system may be a mixture of random variables, plus controllable *decision variables* (such as how many fishing vessels to licence, or treatment options for weeds in a crop).
- For each stage, the benefits and costs of the output states and of the chosen decision variables are combined to determine a solution to the *objective function.* This objective function will commonly be some measure of benefit which is either maximised or minimised (revenue or costs respectively). Often, the measure of benefit will be expressed as a net cash flow.
- Since the system is modeled over multiple stages, the benefits achieved in one stage may be to the detriment of benefits at another stage. It is therefore necessary to combine the objective function values across stages to yield a multi-stage solution. This combination across stages is commonly achieved by discounting later stages at some appropriate discount rate, and then adding the discounted future benefits to obtain a "net present value". The net present value (NPV) can be considered as the overall objective function for the multi-stage dynamic system.

Modeling a natural resource system aids decision makers by allowing them to test the impact of alternative policies. The analyst therefore needs to identify the state variables, an objective function and the decision activities that influence the system being modeled.

Given these, at any stage a set of decision variables and input state variables will generate a set of output state variables. This transition system (from input to output state) also needs to be identified.

The problem can become very complex because, for each possible set of input state variables at each stage, the set of decision variables can produce a different set of output state variables, which in turn are the input state variables for the next stage. One approach to this problem is to use dynamic programming (DP). A clear introduction to DP is provided by Kennedy [1986].

1.1 Dynamic Programming

Consider a system with a finite number of stages, and with state and decision variables that can each take on a finite number of discrete values. We shall refer to a set of state variable values as the system state.

The dynamic programming procedure comprises two parts. First it works backwards from the last stage to the first, identifying at each stage the optimum decisions for each possible starting state. Then the procedure works forward from the starting state of the first stage to the last stage, identifying the optimum sequence of decisions.

Beginning with the last stage, we apply the transition system to identify the optimal decisions to be made for each possible starting state. An objective function value can thus be ascribed to each possible starting state of the last stage. Discounting by one period, we thus have the optimal objective function value and decision for each possible ending state of the previous stage.

We then turn our attention to the second last stage. For each possible starting state we again identify the optimal decisions. This time, we include in the objective function the discounted value of the ending state (brought back from our evaluation of the last stage).

The procedure is repeated for each previous stage, evaluating the optimal objective function value for each possible starting state, which is then discounted by one period to provide the objective function value for each possible ending state of the previous stage.

When we arrive at the first stage, we find we now have the objective function value for each possible starting state of the first stage. Identifying the actual starting state of the first stage, we have found its total net present value, the optimum decision to make in the first stage, and therefore the end state of the first stage. This corresponds to the optimal starting state of the second stage. We can thus continue forward stage by stage, reconstructing the optimal state and decision history that will yield the best achievable net present value.

Some idea of the benefits (and limitations) of DP can be achieved by considering the dimensionality of the problem.

Consider a system with "T" stages, "N" possible states at the beginning and end of each stage, and "M" possible combinations of decision variables at each stage. The total number of possible decision policies over the T stages is M^T, known as total enumeration. However using backwards incursion, a feature of DP, the number of policies to be evaluated for each stage is NM. By the time we have worked back though the T stages, the DP will have evaluated TNM policies. If T, N, and M were each 10, the DP would require evaluation of 1,000 out of the possible 10^{10} policies, reducing the search field by a factor of ten million. However, it should be noted that if T, N and M are large, the product TNM can still be enormous - the "curse of dimensionality".

If the state or decision variables are continuous, the curse of dimensionality is potentially infinite, though various treatments of continuous variables have been proposed. The use of interpolation to deal with the curse of dimensionality will be considered later in this paper.

This paper will consider the application of dynamic programming (DP) to the optimisation of multi-stage natural resource systems, and report upon some work applying graphical user interface (GUI) techniques through a simulation package to simplify and standardise the use of DP. The approach will be illustrated by a case study.

2. METHODOLOGY

Each natural resource system is likely to differ in the nature of its state and decision variables, and in the transition of the states between stages. This has required previous applications of DP to be problem and coding specific.

Usually, the transition system is independent of the stage. One set of decisions applied to an input set of state variables yields a new set of output variables, for each stage. Also, the transition system is much easier to code than is the DP process. If the user could concentrate on specifying the transition system without having to worry about the DP optimisation, the application of DP could be greatly simplified.

However, it would be tedious and inefficient to have them as separate programs, where the output of one had to be fed as an input to the other.

A graphical user interface (GUI) provides a means of decoupling the transition system from the DP optimisation while keeping them integrated in the one modeling environment.

2.1 Modular Representation

Our approach has been to implement the DP algorithm using the GUI simulation package Extend™, enabling us to model the transition system and the DP optimisation as two interconnected modules.

Extend™ is designed for building simulation models as interconnecting blocks, linked by flows. The logic within each block can be programmed in a dialect of "C", and animations can easily be displayed. The simulation features of Extend are particularly suited to the characteristics of transition systems in DP problems.

Each block in Extend has an "icon", or block picture, with flow connectors. Double clicking on the block icon causes it to open up to show a "dialog window", which displays parameters that can be set. Results can also be displayed in the dialog window.

Each block also has a "structure window" which can be opened up to examine or edit its program code.

We used two Extend™ blocks to model the transition system and the DP optimisation, as shown in Figure 1.

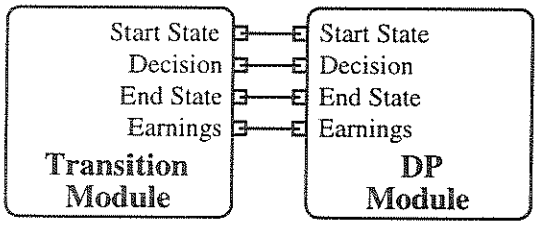

Figure 1: The Dynamic Programming Model

The interconnections allow vector information to flow between the modules. When the DP Module requires an evaluation, it sends the "Start State" and "Decision" vectors to the Transition Module. The Transition module uses these inputs to compute the "End State" vector and the "Earnings", and returns these quantities to the DP Module.

The Transition Module is problem specific, and can be coded (or the Extend module programming capacity can be used, removing the requirement for language specific coding skills) in the language C to represent the appropriate state transition for any particular problem, or problem type. If desired, animation can be added to illustrate the relationships between the state and decision variables. Within a problem class, the module's dialog window can be opened to set relevant parameters controlling the transition.

The DP module is generic, and does not need to be coded for each problem. The module has a dialog window which can be opened to set the dimensionality of the state and the decision vectors, and the possible discrete values they can take. It also allows an output text file to be specified for the simulation results.

3. CASE STUDY - WILD OATS IN WHEAT

The problem of wild oats in wheat has been analysed by a number of methods and levels of sophistication (referred to in Jones and Medd 1997). The DP method developed in Jones and Medd [1997] incorporates summer and winter crop activities and winter fallow. The decision alternatives include leaving the land in winter fallow, and planting sequences of sorghum or wheat. The wild oats may be subject to management options of no control, plant kill, seed kill, or both plant and seed kill.

The critical state variable determining the weed population is the wild oat seed bank. In a single generation, some seeds germinate to seedlings, of which a proportion survive to maturity and deposit seed to the seed bank. To capture the asynchronous germination of wild oats through a season, three cohorts are simulated each year (or stage) of the model. The residual seed remains dormant with a proportion carrying over to germinate in the following year.

Plant survival and seed survival depend on the wild oat management options of no control, plant kill, seed kill, or both plant and seed kill.

Using the model of Jones and Medd [1997], for each stage a decision must be made as to which of fifteen possible treatment options to use. There are also two state variables to be taken into account, the state of the land (ascribed fifteen possible values) and the density of the seed bank (ascribed 5,000 possible values).

Figure 2 shows the dialog window of the transition module, in which can be entered the parameters needed for the transition computations of the Jones and Medd [1997] model.

Wheat Price	133	Sorghum Price	139
Wheat Variable Cost	118.38	Sorghum Variable Cost	154.04
Yield Wheat1	3.5	Yield Sorghum1	3.75
Yield Loss Wheat2	0.2	Yield Sorghum2	1.8
Yield Loss Wheat3	0.4	Delta	104.4
Control Costs:-		Theta	1.22
Winter Fallow	31.08	Gamma	-1
Herb Seed	19.31	Germinate	0.5
Herb Plant	25.4	Decay Rate	0.75

Control Parameters

Row	pm_c1	pm_c2	pm_c3	ss_c1	ss_c2	ss_c3
0	0.85	0.15	0.75	0.6	0.6	0.6
1	0.85	0.75	0.75	0.5	0.5	0.5
2	0.85	0.15	0.5	0.2	0.2	0.2
3	0.85	0.75	0.75	0.2	0.2	0.2
4	1	1	1	0	0	0

Plant Kill Parameters

Row	Alpha	Lambda	Epsilon
0	8.6	0.74	0.88
1	7.6	1.2	0.8
2	6.8	2	0.67

Cohort Composition

Row	% (=1)
0	0.3
1	0.6
2	0.1

Seed Kill Parameters

Row	Alpha	Lambda	Epsilon
0	7.42	2.04	0.66
1	7.42	2.04	0.66
2	7.42	2.04	0.66

Phytotoxicity Seed Kill	0.02
Phytotoxicity Plant	0.01

Figure 2: Transition Module Dialog

The Transition Module is programmed to compute the end state values and earnings for a stage, given the input starting state values (seed bank, land history) and the decision values (weed treatment). The inputs are received as vectors from the DP Module, and the computed end state vector and earnings are outputs transferred back to the DP Module.

Figure 3 shows the dialog box for the DP Module. The problem can be named, and an output text file nominated to record the results of the simulation. The example shown is to be computed for ten stages (years) and the earnings are to be discounted at a rate of 10% per annum.

The DP Module is programmed to cope with problems having up to five state variables and up to five decision variables. This example uses only two state variables (15 and 5,001 values respectively) and one decision variable (with 15 possible values).

It would of course be equally feasible to reformulate the problem with a greater number of state and decision variables, each having a lesser number of possible values. Initial values of the state variables are also set in this window.

Dynamic 2 Problem Name

OutEig.txt Output File Name

◉ Create New File

○ Append to Existing File

10 Number of Decision Stages

10 % Discount Rate

State Variables - State[0]...[4]

Row	Minimum	Maximum
0	0	14
1	0	5000
2	0	0
3	0	0
4	0	0

Row	Grid Interval	Initial Value
0	1	0
1	1	5000
2	1	0
3	1	0
4	1	0

Decision Variables - Dec[0]...[4]

Row	Minimum	Maximum
0	0	14
1	0	0
2	0	0
3	0	0
4	0	0

Figure 3: DP Module Dialog

To explore the effects of the trade-off between accuracy and the curse of dimensionality, each state variable can be ascribed a grid interval. For example, a state variable with values ranging from 0 to 100, and a grid interval of 5, would have possible values of 0, 5, 10 ... 95, 100. Thus the matrix dimensions can be reduced by increasing the grid size.

The output file generated by the simulation is shown in Figure 4. It shows that the optimum earnings achievable have a discounted net present value of $1,190, and are achieved by applying decision 1 in years 0, 2, 4 and 6, alternating with decision 0 in years 1, 3, 5 and 7. Finally, policies 3 and 7 should be applied in years 8 and 9.

It is of interest to note that maximising the ten-year net present value requires losses be made in years 1 and 3.

The end states for each stage (year) are not shown, since they are the beginning states for the following year. The final state of year 10 is displayed, in order to give the beginning state of the eleventh year, at the end of the period computed. No decisions or earnings are shown for this eleventh year, since the planning horizon for this run is ten years.

4. DISCUSSION AND CONCLUSIONS

We have considered the use of a graphical user interface simulation package to decouple the transition system, which tends to be problem specific, from the dynamic programming which we have formulated generically. This decoupling enables the practitioner to specify the transition system and use dynamic programming without having to be involved with a difficult programming task. The method has been illustrated successfully using a real-world agricultural problem.

Some problems remain to be considered. In particular, the method depends upon state and decision variables being of discrete rather than continuous values. If too many values are possible, then the curse of dimensionality sets in.

One approach to the curse of dimensionality is to use interpolation. A fairly coarse grid of state and decision variable values can be constructed. Starting with the coarse grid decisions and starting states, computed end states may be found to lie between the grid values. In carrying out the forward induction, these intermediate end state values can be ascribed earnings and subsequent decision histories interpolated from their neighbouring grid values. While superficially attractive, this approach assumes that the relationship between state and optimum decision policy is continuous. This assumption needs to be tested on a natural resource system before it is applied. One way to do this has been envisaged in our model design. By repeating runs with the grid interval set successively finer (see figure 3), inconsistency in results would be indicative of important discontinuities in the relation between the initial state and the optimum decision policy.

5. REFERENCES

Kennedy, J.O.S., Dynamic Programming: Applications to Agriculture and Natural Resources, Elsevier, 341pp., London, 1986.

Jones, R., Medd, R., The Economics of Weeds Control Strategies: A Dynamic Programming Analysis of Wild Oats Control, Paper presented at the Australian Agricultural and Resource Economics Society Broadbeach, January 20-25 1997.

Optimal Path for Dynamic Program, EIGEN "Dynamic2" Run, 3:49:52 PM, 28/7/97												
Stages		States		Decisions		Discount Rate						
10		75015		15		10%						
	State	State	State	State	State	Decis.	Decis.	Decis.	Decis.	Decis.		
Min	[0]	[1]	[2]	[3]	[4]	[0]	[1]	[2]	[3]	[4]		
Max	0	0	0	0	0	0	0	0	0	0		
Incr	0	0	0	0	0	0	0	0	0	0		
Begin	State	State	State	State	State	Decis.	Decis.	Decis.	Decis.	Decis.	Earn	Earn
Stage	[0]	[1]	[2]	[3]	[4]	[0]	[1]	[2]	[3]	[4]	Stage	Cumul.
0	0	5000	0	0	0	1	0	0	0	0	$333	$1,190
1	1	5000	0	0	0	0	0	0	0	0	-$28	$943
2	0	706	0	0	0	1	0	0	0	0	$333	$1,068
3	1	706	0	0	0	0	0	0	0	0	-$28	$808
4	0	99	0	0	0	1	0	0	0	0	$333	$920
5	1	99	0	0	0	0	0	0	0	0	-$28	$645
6	0	13	0	0	0	1	0	0	0	0	$333	$741
7	1	13	0	0	0	0	0	0	0	0	-$28	$448
8	0	1	0	0	0	3	0	0	0	0	$315	$524
9	3	0	0	0	0	7	0	0	0	0	$230	$230
10	7	0	0	0	0							

Figure 4: The Output File