AN ABSTRACT OF THE THESIS OF

Gonzalo L. Paredes-Veloso for the degree of Master of Science in Forest Management presented on March 10, 1986.

Title: An Efficient Formulation and Solution Method of the Even-aged Rotation and Thinning Problem.

Abstract approved: Signature redacted for privacy.

J. Douglas Brodie

The problem of determining an optimal sequence of decisions in even-aged stand management is analyzed and solved with an alternative to the traditional dynamic programming network. The new method is derived independently from two different approaches to the problem: Network optimization and generalized LaGrange multipliers. Both derivations are presented. It is also shown that the new method could have been derived from application of the discrete-time Maximum Principle of control theory.

An illustrative example is presented and the computational efficiency of the new method is demonstrated by comparing its performance with an existing dynamic programming model.

The aspects of the new solution method requiring further analysis and research are stated.

Keywords: dynamic programming, LaGrange multipliers, optimal control, network optimization, even-aged management.

An efficient specification and solution method of the even-aged rotation and thinning problem

by

Gonzalo L. Paredes-Veloso

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Completed March 10, 1986

Commencement June 1986

ACKNOWLEDGEMENTS

I acknowledge my indebtness to the Faculty, staff and graduate fellows of the Department of Forest Management. The environment of excelence they have created and maintain has been a permanent stimulus during my graduate studies. I am thankful to Professor Douglas Brodie who encouraged me to undertake this topic, read a large chunk of the manuscript and provided detailed comment and boundless enthusiasm when it was badly needed. His interest, advice and criticism make me deeply indebted.

My gratitude to my wife, Ana Maria, who is my permanent source of encouragement. I dedicate her this work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

AN EFFICIENT FORMULATION AND SOLUTION METHOD OF THE EVEN-AGED ROTATION AND THINNING PROBLEM

INTRODUCTION

The problem of determining the optimal sequence of management decisions for a stand has been one of the most frequently discussed topics in forest management during the last two decades. The development and spread of modern mathematical optimization tools allowed an early theoretical discussion of the problem showing that dynamic programming (Arimizu, 1958; Amidon and Akin, 1968) and optimal control (Naslund, 1969) were suitable mathematical approaches to its solution. During that time the discussion was constrained to theoretical relations between the mathematical approach and economic and silvicultural concepts traditional in forestry. Solution methods were lacking in most of those reports, and when present they dealt with simplistic situations.

The solution of practical problems came in the late 70's largely estimulated by an increasing availability of electronic data processors and the development of stand growth simulators capable of handling the effects of silvicultural prescriptions. Brodie and Kao (1979) reported solutions using an existing intensive management simulator and the dynamic programming technique. Since then, the technique has been applied to numerous stand models with a wide range of decision alternatives (Brodie and Haight, 1985).

However, it is recognized that the current approach for dynamic programming solution suffers the "curse of dimensionality" when applied to practical problems requiring several state variables (Hann and Brodie, 1980), and therefore its applicability is restricted either by computing hardware capacity or by computing budget.

The search for new approaches to solve the problem has led to the development of the solution method reported in the next sections. As will be shown, the method has been derived from the application of a combinatorial technique, and further explained by using other available optimization techniques.

The method is next illustrated with a simple example, and finally its efficiency is demonstrated by comparing it with an existing dynamic programming algorithm.

STATEMENT OF THE PROBLEM

The problem of optimizing a sequence of management decisions in even-aged stands can be described in a number of ways; actually the literature shows a wide range of formats and notations depending upon the corresponding approach to solve it. For the purpose of this paper it is described as a discrete-time control problem since it embodies the dynamic allocation of scarce resouces among competitive activities over a time interval. Even though the passage of time is continuous a discrete-time version has been choosen given the focus on practical implementation of the methods analyzed and proposed: activities are rather considered as undertaken at points in time during the growth of stands.

The problem is therefore a multistage optimization problem that can be described as maximizing, over T_n

$$f_N(Y_N) = \sum_{n=1}^{N} r_n(T_n) \tag{1}$$

subject to:

$$X_n - T_n + G_{n+1}(Y_n) = X_{n+1} \qquad (n=1,..,N-1) \tag{2}$$

$$X_n - T_n = Y_n \qquad (n=1,..,N) \tag{3}$$

$$X_N - T_N = 0 \tag{4}$$

(alternative constraint for (2):)

$$Y_n + G_{n+1}(Y_n) - T_{n+1} = Y_{n+1} \qquad (n=1,...N-1) \tag{5}$$

where n : represents a stage or growth period. It is directly related with stand age. N is the final stage.

Y_n : vector describing the stand at stage n after a decision $[T_n]$ has been undertaken in a stand described by X_n at that stage. It is often called the residual stand vector, state descriptor or state variable.

$f_N(Y_N)$: objective function value of a management N-decisions sequence yielding a stand described by Y_N. In the context of this paper Y_n corresponds to clearcut at stage N.

X_n : vector describing the stand at stage n after it grows from its state Y_{n-1}. It is sometimes referred to as the initial stand vector at stage n.

T_n : vector of feasible management decisions at stage n transforming the stand X_n into Y_n and generating a return $r_n(X_n, T_n)$.

$r_n(T_n)$: return generated at stage n when a decision T_n has been taken

$G_{n+1}(Y_n)$: growth of a stand described by Y_n at the stage n to the next stage n+1. It is a function transforming Y_n into X_{n+1} which together with T_n constitutes the transformation function as commonly presented in traditional dynamic programming textbooks (Nemhauser, 1966).

Equation 1 is commonly refered to as the objective function and shows that the solution values depend on the values collected along the "path" from stage 1 to stage N. In control theory this equation is referred to as the objective functional and it is normally presented as decomposed into one term describing the sum (integration) of revenues along the path itself, and another term describing the effect of terminal conditions.

Equation 2 is known as the state equation since it links state variables, decision variables and time. By transfering X_n to the left-hand side this equation becomes the discrete-time analogue of the "motion equation" of control theory. This equation can be further compressed into functional notation, however the form chosen here will allow for a clearer understanding of its meaning.

Equation 3 states that any change in the state of the stand, within a stage, has to be derived from implementing a decision option. This consistency constraint normally relies on the stand growth simulator being utilized. In other words, the transformation functions to handle the effect of decision variables and to generate state variables should be available.

Equation 4 simply describes the state of the stand after stage N: clearcut. The time at which this final state is reached remains unspecified thus being also a decision variable.

A problem is then fully specified with the above equations and given the initial stand conditions and the feasible space for decision and state variables.

The boundaries of the feasible space are normally congruent with boundaries embedded in the stand growth simulator. Particular features of actual problems may suggest additional restrictions on the feasible space, however in no case can this exceed that of the stand simulator.

Approaches to solution

This section describes some of the approaches already reported to solve the complete version of the problem. Boundaries are traced according to the treatment given to the time variable and to constraints imposed in the feasible space.

The treatment of time permits differentiation into two classes of problem approaches. The first approach relies on properties of the production surface and solution space but not on the passage of time from initial states through transitional states to steady state. This is characterized as a static optimization approach. The second approach relies upon the dynamic nature of the problem thus giving a special role to the passage of time: among these approaches are the classical calculus of variations, dynamic programming and the maximum principle. The following description of approaches refers particularly to the above.

The description is also centered on those approaches reported to solve the complete specification of the problem. Therefore attention is not given to restricted versions such as those with an empty feasible set for intermediate decisions (from stage 1 to N-1), since in this situation the problem becomes one of solving only for the optimum rotation age. Also discarded is the version where the set of intermediate decisions is so small that a brute force approach can be easily applied by evaluating all the options.

Classical Calculus of Variations

The classical calculus of variations as a solution approach has been reported only by Cawrse and others (1984). In essence the approach is the dynamic analogue of the classic static programming problem: the state equation is substituted into the objective function and necessary conditions analogous to the first order conditions that the derivative vanish lead to the Euler equation (Intriligator, 1971).

Despite its analytical elegance, the method presents certain weaknesses when practical implementation is considered. First, it imposes strong restrictions on the mathematical properties of the functions modeling stand growth and control variables. These restrictions are certainly consistent with biological and economic theory, however the actual trend in forest modeling is characterized by stand simulators composed of a large number of interrelated functions that discourage a calculus-based approach to solution. As shown in Cawrse and others (1984) the successful application of the method depends on tailored and simple production functions. Secondly, the method does not allow handling of more constraints than the state equation. And finally the treatment of the rate of thinning variable - and in general the rate at which the control variable is applied - as a continuous differentiable function of time does not reflect actual operation conditions in stand management problems.

The Maximum Principle

The Maximum Principle approach (Intriligator, 1971) was early proposed by Naslund (1969) as possible approach to solve the problem, however he did not report a solution method. In general the method can be considered as an extension of the method of LaGrange multipliers to dynamic optimization problems. As in static problems a row vector of new variables, the adjoint or costate variables, is added to the set of constraints. These new variables are the dynamic analogues to the LaGrange multipliers of static optimization problems with a contraint set.

The approach, as in classical calculus of variations, requires all functions involved to be nonzero continuously differentiable functions of time.

The method also has analytical elegance but the practical application of the continuous-time version is also restricted to simple situations. The Maximum Principle approach has been later reported by Sethi (1973), McDonough and Park (1975), and recently by Donnelly and

Betters (1985).

Dynamic programming

In the late 60's the dynamic programming approach began to be proposed as a solution method to the discrete-time version of this problem (Amidon and Akin, 1967). Its suitability has been shown since then with a variety of growth models and decision variables, especially during the last decade. It has characteristics and capabilities that provide a useful solution framework for timing and intensity of thinning, fertilization, pest control, and rotation age decisions. Also the approach can generally handle any restriction in the control set by construction of the recursive algorithm.

Early papers described the basic approach in stand management problems for simultaneous determination of thinning regime and rotation age solving with the backward recursive method (Amidon and Akin, 1967; Schreuder, 1971); however, the growth models used were primitive and simplified so the approaches were not extended or adapted to other problems. Brodie and Kao (1979) implemented a useful approach utilizing an existing stand growth simulator for Douglas-fir, DFIT (Bruce and others, 1977), and a forward recursion to solve simultaneously for thinning regime and rotation age. Their formulation has been, since then, extended and adapted to solve for other decision variables and species, such as grazing and thinnings for ponderosa pine (Riitters and others, 1982), diameter structure and thinnings for lodgepole pine, (Haight and others, 1985), hardwood release and thinnings for loblolly pine (Valsta and Brodie, 1985), and for thinnings and rotation age in red pine stands (Martin and Ek, 1981).

Analysis of approaches

As pointed out by Intriligator (1971) the Maximum Principle provides a more general solution approach to solution of the control problem than dynamic programming. However in practical applications the emphasis has shifted from analytical procedures to numerical techniques applied to the discrete-time version of the problem. Two main reasons can be pointed out to explain this: real situations in stand-level management are often analytically intractable and a tailored production surface has to be specially developed if calculus-based procedures are to be applied. Secondly the availability of faster and better computers has, in general stimulated the development of combinatorial algorithms and particularly the usage of dynamic programming as an optimization approach for stand-level management problems.

However, limitations in the application of the dynamic programming approach arise when the problem involves several state descriptors and decision variables. The systematic search implicit in existing dynamic programming algorithms results in large storage requirements for state descriptions and therefore in massive iterative computations as the dimensionality of the problem increases.

As pointed out by Hann and Brodie (1980) the approach currently utilized for the dynamic programming (DP) formulation of stand management decision presents limitations with regard to computational efficiency for complex stand simulators and as a result, computing cost, or time may become restrictive using either mainframe or personal computers.

The approach currently utilized in dynamic programming models has been described in many textbooks. A description of the corresponding algorithm applied to stand-level problems has been presented in Paredes and Brodie (1985).

The cause for the main drawback of the traditional formulation is that it requires as many state descriptors as there are constraints relating state and decision variables in the canonical form. In practice the number of state descriptors is often more, depending upon the complexity of the stand growth model embedded in the DP algorithm. As pointed out by Hann and Brodie (1980) the approach would become impractical when applied to whole stand/diameter class models, having number-of-trees per diameter class as state descriptors. Generally the canonical form has one or more types of constraints relating all the decision variables and state descriptors. Furthermore the transformation function (stand growth model transforming initial state and decision in final state) may require one or more additional state descriptors to adequately represent the response surface. Brodie and Haight (1985) refer to these extra state descriptors as storage variables.

Therefore the number of elementary calculations required by the systematic search in the traditional algorithm increases exponentially as the number of state descriptors increases. Even in modern mainframe computers the execution time involved restricts the range of practical applicability for the method. The usage of microcomputers becomes even more restricted to the solution of oversimplified networks.

Another characteristic of this formulation is that when searching forward for the optimum path to each state it uses the cumulative return as constituted by all revenues from previous silvicultural and stocking decisions without including any value for the standing trees, except at the final harvest node. However information describing the residual stand is recorded through the set of state descriptors.

It will be demonstrated that this widely applied algorithm, whose feasibility has been demonstrated on a number of growth models is comparatively inefficient, from a computational viewpoint, especially when solving decision sequences with a growth model capable of dealing with several decision variables and requiring several state descriptors. On the other hand, this algorithm is general, capable of dealing with a high degree of nonconcavity in production and discontinuity in cost or revenue functions.

BASIS OF THE PROPOSED SOLUTION METHOD

The proposed algorithm can be derived either from a network formulation of the problem, or from generalized LaGrange multipliers theory, and even from the necessary conditions of the Maximum Principle. This section presents the two first bases and also analizes the analogies with a discrete-time version of the Maximum Principle.

A Network Approach

The proposed algorithm was derived from an appropriate application of a commonly used algorithm, developed by Dijkstra (1959), to solve for the shortest-path in networks. The algorithm is applied to a network describing stand development under different management decisions.

Dijkstra's algorithm has been described in many textbooks of operations research and consequently its formulation can be avoided in the present paper. For description and examples Hu (1982), and Hillier and Lieberman (1980) are suggested as references. For forestry applications Dijkstra's algorithm is described by Dykstra (1984).

This shortest-path algorithm searches for the path of minimum distance between a pair of nodes in a network without negative cycles. A cycle in graph theory is a path with all the nodes distinct except the first and the last, which are the same node. Thus if the network is acyclic with positive or negative arcs the algorithm can be slightly modified and applied to search for the longest path in the network (Hu, 1982, sections 1.2 and 1.5). Therefore the equivalent maximization problem results in

$$\max(f) = -[\min(-f)]$$

and the problem of maximizing returns from a management regime can be stated as the search for the longest route in a network describing the stand development and decisions.

Figure 1 describes a segment of such a network for the n-th stage. The nodes are clustered in stages, each stage representing a growth period, and at each stage a node represents a candidate decision to be taken. Obviously this is an acyclic network.

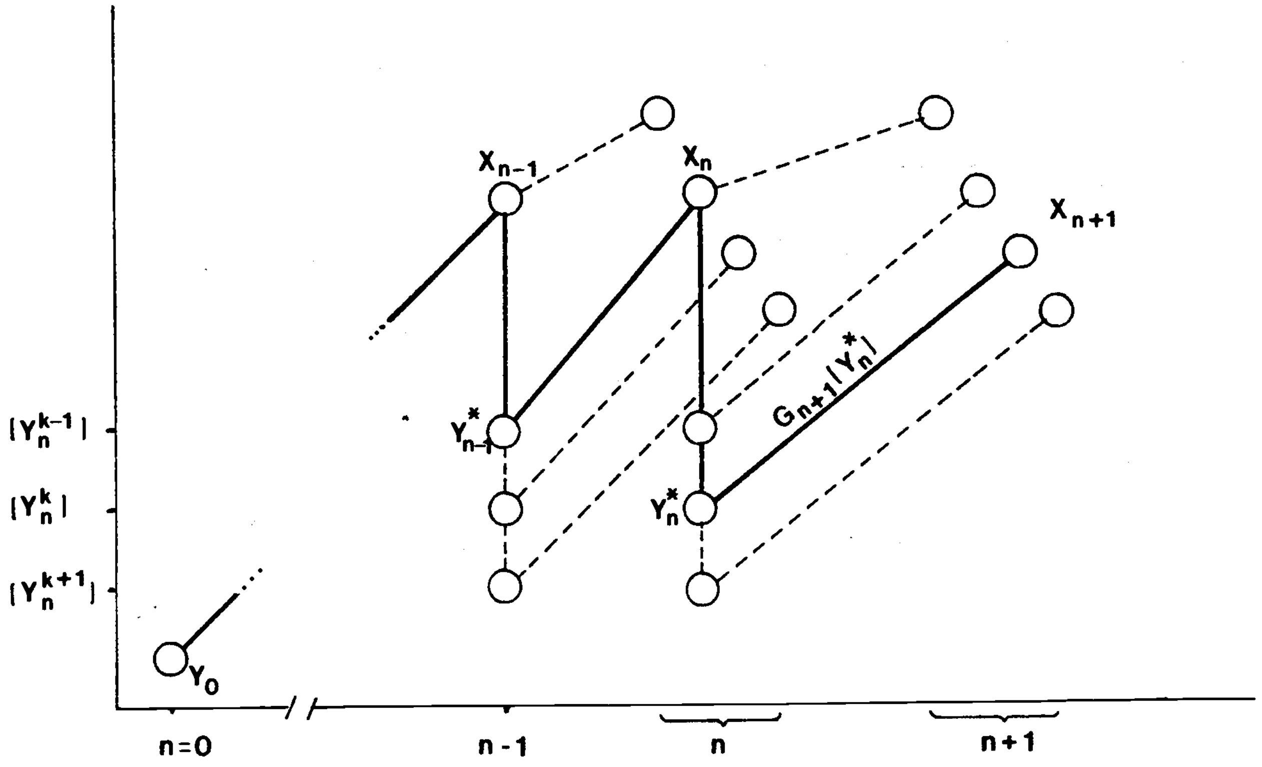

Figure 1. Interval [n-1,n+1] of a network describing stand management options. Full lines are arcs incorporated in the objective function. Dotted lines are arcs considered in recursion.

At each iteration, the algorithm examines those nodes with a temporary label (a temporary label "h" is defined to be the length of a path from the origin to that node, so it is a lower limit to the longest path) that are linked with the set of nodes permanently labeled (a permanent label "h^*" is the true longest distance from the origin to that node). In Figure 1, the set $[Y_{n+1}]$ whose elements are linked directly with the set $[Y_n]$, already permanently labeled is examined. We evaluate the distances between each element of $[Y_n]$ and $[Y_{n+1}]$; theoretically it would be necessary to evaluate K^2 distances, but in practice no more than K distances need be evaluated since

$$d(Y_n^k, Y_{n+1}^1) = d(Y_n^k, Y_{n+1}^k) \quad \text{for } k=1,\dots,M \tag{6}$$

where K is the number of nodes at stages and $d(Y_n^k, Y_{n+1}^k)$ represents the "distance" between a stand described by Y_n^k at stage n and the node described by Y_{n+1}^k at the next stage.

In words this means that it does not matter which decision is to be taken in the next stage since the stand will grow the same; what does matter is the decision being taken at the present stage.

The purpose of examining $[Y_{n+1}]$ is to search for the node having the maximum cumulative distance from the starting node. Accept for a moment that the cumulative distance for all nodes in the set $[Y_n]$ is the same, then the search is for

$$h^*_{n+1}(Y^k_{n+1}) = \max_{[T_n]} [r_n(T_n^k) + G_{n+1}(Y_n)] \tag{7}$$

where $h^*_n(Y^k_n)$ is the permanent label equal to the maximum revenue generated by the decision T_n^k and the growth of a stand having state Y_n after the decision T_n^k is implemented, $G_{n+1}(Y_n)$.

Since it was accepted (for the moment) that the cumulative distance for all nodes in the set $[Y_n]$ was the same, then all nodes in the set $[Y_{n+1}]$ are going to receive the same permanent label equal to

$$f_{n+1}(Y^k_{n+1}) = h^*_{n+1}(Y^k_{n+1}) + f_n(Y^k_n) \qquad (8)$$

where $f_n(Y_n)$ is the cumulative distance up to stage n and the pair [K,n] identifies the node for optimum $h^*_{n+1}(Y^k_{n+1})$.

So far the reader has been forced to accept that at stage n the cumulative distance $f_n(Y^k_n)$ is the same for all k. By using inductive reasoning from stage n=1 it is observed that, for Y_0 representing the initial state of the stand, the growth of the stand up to the stage n=1 is necessarily the same, so $f_1(Y^k_1)$ is a constant for all k in the first state. Now the search is in the set $[Y_2]$ and repeats the same steps to assign a permanent label as described above. The proof is thus completed.

Having already described the general iterative procedure, it is necessary to point out that the algorithm analyzes and solves completely one stage at each single iteration. Thus it can be generalized that it requires as many iterations as stages times the number of nodes at each stage in the network. This is an upper limit because, as shown, at each iteration the set of nodes at the next stage $[Y_{n+1}]$ becomes permanently labeled with the same value (i.e. cumulative return) and all of them share an unique nexus with the previous stage: that node yielding the maximum value of the objective function after solving the current stage.

Two main outcomes arise from the above procedure; first, the algorithm presents a tremendous increase in efficiency in terms of code implementation and computing time when compared with traditional DP formulations to solve the same problem, as will be demonstrated in the following sections. Second, it is unnecessary to utilize the state variables of the growth model as potential state descriptors for storing the residual stand in a multidimensional network after each decision has been undertaken. This also makes the algorithm more easily implementable.

The basic functional equation for the proposed approach is:

$$f_n(Y_n) = \max_{[T_n]} [r_n[T_n, G_{n+1}(Y_n)]] + f_{n-1}(Y_{n-1}) \tag{9}$$

where $r_n[T_n, G_{n+1}(Y_n)]$ represents the "net revenue" to be added (or deleted, depending upon its sign) to the previous value of the functional equation. For example, if the objective were to maximize the total volume harvested from the stand, equation 9 would read, in volume units,as

$$f_n(Y_n) = \max_{[T_n]} [T_n + G_{n+1}(Y_n)] + f_{n-1}(Y_{n-1}) \tag{10}$$

with $f_{n-1}(T_{n-1})$ representing the accumulated volume up to stage n after deciding to leave a residual stand Y_{n-1} at the previous stage . T_n is the harvested volume in the current stage; $G_{n+1}(Y_n)$ is the volume growth from a stand having a residual volume Y_n.

When implementing the algorithm to solve for an economic criterion (present net worth, or soil expectation) the functional equation remains the same but it requires some simple transformations to account for revenue and cost functions, interest rate and inflation. In these cases, the appropriate form of the functional equation is

$$f_n(Y_n) = \max_{[T_n]}[r_n(T_n)+r_{n+1}(X_{n+1},T_n)]-r_n(X_n,T_{n-1})+f_{n-1}(Y_{n-1}) \tag{11}$$

where $f_n(Y_n)$ represents the accumulated present value of previous decisions including the value of the residual stand Y_n after it grows to stage n+1.

The term $r_n(X_n, T_{n-1})$ is the value of the residual stand Y_{n-1} left at the previous stage n-1 and since it is already included in $f_{n-1}(Y_{n-1})$ it has to be substracted and replaced by $r_{n+1}(X_{n+1}, T_n)$.

It is noted, in equation 11, that the composite term

$$r_{n+1}(X_{n+1}, T_n) - r_n(X_n, T_{n-1})$$

accounts for the growth, in monetary units, of the residual stand after the decision T_n has been undertaken. It is the analogous concept to $G_{n+1}(Y_n)$ in equation 10.

According to the above concepts and definitions the new algorithm can be summarized as an iterative procedure involving the following steps:

```
Step 0   $f_0(T_0) = r_0(X_0) + R$
         $n \longleftarrow 1$
Step 1   $P_n(T_n) \longleftarrow r_n(X_n, T_n) + r_{n+1}(X_{n+1}, T_n) - r_n(X_n, T_{n-1})$
         $f_n(Y_n) \longleftarrow \max_{[T_n]}[P_n(T_n) + f_{n-1}(Y_{n-1})]$
         Save $T_n^*$
Step 2   $f_n(Y_n=0) < f_{n-1}(Y_{n-1}=0) \Longrightarrow$ STOP, PRINT $[T^*]$
         $n \longleftarrow n + 1$
         $X_n \longleftarrow Y_{n-1}^* + G_n(Y_{n-1})$
         GO TO Step 1
```

The first statement of step 2 provides the stopping rule when the maximum value of the objective function has been surpassed. Since local maxima may exist when determining rotation age with an economic criteria, for example a christmas trees or pulpwood rotation the stopping rule has to be designed considering such a possibility.

Lagrange Multipliers approach

The algorithm already described can also be derived from the canonical form of the problem (equations 1 through 4) by using LaGrangian multipliers in those constraints relating state and decision variables. This procedure is straightforward for resolving the main drawback of the traditional DP formulations derived directly from the canonical form: its high dimensionality in the state space.

Equation 2 describes those constraints. Using a LaGrangian multiplier for the left-hand side of each constraint allows them to be shifted to the objective function and thus eliminates the associated state descriptors. This procedure to eliminate constraints was initially proposed by Dreyfus (1957), later extended in dynamic programming by Bellman and Dreyfus (1962), and generalized as a method to solve problems of optimum allocation of resources by Everett (1963).

The objective function in canonical form becomes:

$$\max_{[T_n]} f_N(Y_N) = \sum_{n=1}^{N} r_n(T_n) - \sum_{n=1}^{N} \lambda_n[X_n - T_n + G_{n+1}(Y_n)] \tag{12}$$

where λ_n is the LaGrange multiplier, shadow price or opportunity cost of the resource (X_{n+1}) constraining the solution space; and the functional equation of the dynamic programming problem associated with the modified canonical form becomes:

$$f_n(Y_n) = \max_{[T_n]}[r_n(T_n) - \lambda_n[X_n - T_n + G_{n+1}(Y_n)]] + f_{n-1}(Y_{n-1}) \tag{13}$$

The problem has been reduced in dimensionality of the state space at the cost of introducing one decision variable for each dimension eliminated from the state space of the original problem. The search is now for the pair (T_n, λ_n) that maximizes the objective function and satisfies the constraint eliminated.

It is noted that the relation between the term $\lambda_n[X_n-T_n+G_{n+1}(Y_n)]$ in equation 13 and the composite element $[r_{n+1}(X_{n+1},T_n) - r_n(X_n,T_{n-1})]$ from equation 11 becomes straightforward. Both terms represent increments or decrements to the objective function value at the previous stage. The second one presents the advantage of being easily computed.

As described above the algorithm thus analyses and solves completely one stage at each single iteration in the way of Everett's procedure for optimum allocation.

Generally, when the procedure is applied to allocation problems, this technique implies a search for a value of λ_n that satisfies both conditions. Everett (1963) solved multistage allocation problems by searching the values of λ_n that resulted in an optimum allocation while satisfying the constraint of the problem. Methodological aspects of the search have been discussed by Fox (1966), Fox and Landi (1970), and summarized by Denardo (1982). For the optimal stocking problem in stand management the resource being utilized when a decision T_n is taken is the stand available at the beginning of the next stage, X_{n+1}. This is the resource for which the shadow price must be estimated.

Therefore λ_n can be interpreted as the price per unit of resource X_{n+1} the manager is utilizing when deciding T_n. In this way λ_n can be positive or negative implying that a decision T_n means, respectively, a cost or a revenue for utilizing the standing trees, depending upon the reaction of the stand to T_n, and the cost factors.

This interpretation of the multipliers is congruent with silvicultural criteria traditionally utilized for prescribing a sequence of treatments to a stand since each decision is evaluated by considering both the direct return (or cost) from the decision and the return from future productivity of the residual growing stock after the decision is taken. The multipliers thus represent the opportunity cost of modifying the residual stand.

The advantages over the traditional dynamic programming algorithm become obvious: a problem having N stages and K states at each stage requires $[K*(K-1)+1]*(N-2)+2K \approx (K^2)N$ elementary calculations of the transformation function with the traditional algorithm. The proposed approach would require only $[K(N-1)] \approx K*N$ elementary calculations, and therefore reduces the computing time by a factor of K, the number of states. Certainly for small number of states and stages the savings are unimportant, but for a real situation involving M state descriptors each having K possible states the savings increase exponentially, by a factor of K^M.

An interpretation with the Maximum Principle

The Maximum Principle approach to the optimal control problem introduces the Hamiltonian term to define the necessary conditions for an optimum solution to the problem (Intriligator, 1971). The term is composed by two elements: the first accounts for the flow of accumulated dividends to the objective functional, and a second term representing the flow of investment in capital stock times its shadow price (Dorfman, 1969; Clark, 1976). The sum of both terms results in the Hamiltonian, an expresion for the total rate of increase of the total assets. Therefore, by the Maximum Principle, an optimal control must maximize , along the time path, the rate of increase of total capital.

In the proposed algorithm the first two terms on the right-hand side of equation 13 become a discrete time analogue for the Hamiltonian, gn representing the adjoint (costate) variable or the marginal value of the capital stock (residual stand) at stage n as defined by Dorfman (1969). The Hamiltonian is then maximized at each stage by choice of decision variables according to step 1 of the algorithm.

In a control problem the adjoint variable refers to the fact that the asset's value is not its direct sale value, but the value imputed from future productivity presuming that an optimal sequence of decisions is to be utilized in the remaining stages. The algorithm is consistent with the definition since the residual stand is evaluated after it develops to the next stage, when the next set of decisions is to be considered. At this point it could be argued that this procedure does not result in an exact evaluation of future productivity and it can lead to erroneous results when maximizing the Hamiltonian at each stage.

An implicit assumption is embedded in our method: a one-period growth of the residual stand is an unbiased estimate of future productivity as long as the time length of the period allows all the effects of a decision variable to be represented either as direct return from the decision undertaken or as incremental growth in the residual stand. This is not an unknown assumption since it is also required for stage specification in dynamic programming. The recursive structure of all complex growth models also provides a sound base to the above assumption.

The Hamiltonian is therefore maximized by any suitable search technique on the decision space, and even by exhaustive enumeration when it represents an ill-behaved response surface.

AN ILLUSTRATIVE EXAMPLE

The foregoing sections were devoted to a mathematical formulation and analysis of the traditional and new algorithms. A simple example will show how the new algorithm can be used in practice, how it compares with the traditional approach, and how it can be simply implemented. In order to allow for a better understanding of the new algorithm an example is presented using the same data and functions that Johnson and Sleavin (1984) used to explain a traditional DP model they developed to optimize the management of Douglas-fir stands.

The statement for the problem presented by Johnson and Sleavin (1984) can be summarized as follows: a stand is 15 years old and it presents a standing volume of 500 cu.ft.; the ages for possible thinnings are 20, 25, and 30 years; the rotation age has been fixed at age 35 years; the feasible stocking levels at each stage are full stocking (unthinned stand), 1500 and 1000 cu.ft. for all stages except the last when the stand has to be clearcut. Figure 2 describes the stand development for different management options under the traditional DP approach.

Using the notation introduced in previous sections and the data provided by Johnson and Sleavin, the above problem is described by the following vectors

$Y_o = [500]$ $\quad$ $X_o = [500]$

$Y_1 = [1992, 1500, 1000]$ $\quad$ $X_1 = [1992]$

$Y_2 = [3648, 1500, 1000]$ $\quad$ $X_2 = [3648, 3055, 2371]$

$Y_3 = [4337, 1500, 1000]$ $\quad$ $X_3 = [4337, 2545, 1995]$

$Y_4 = [\ 0\ ]$ $\quad$ $X_4 = [4185, 2253, 1778]$

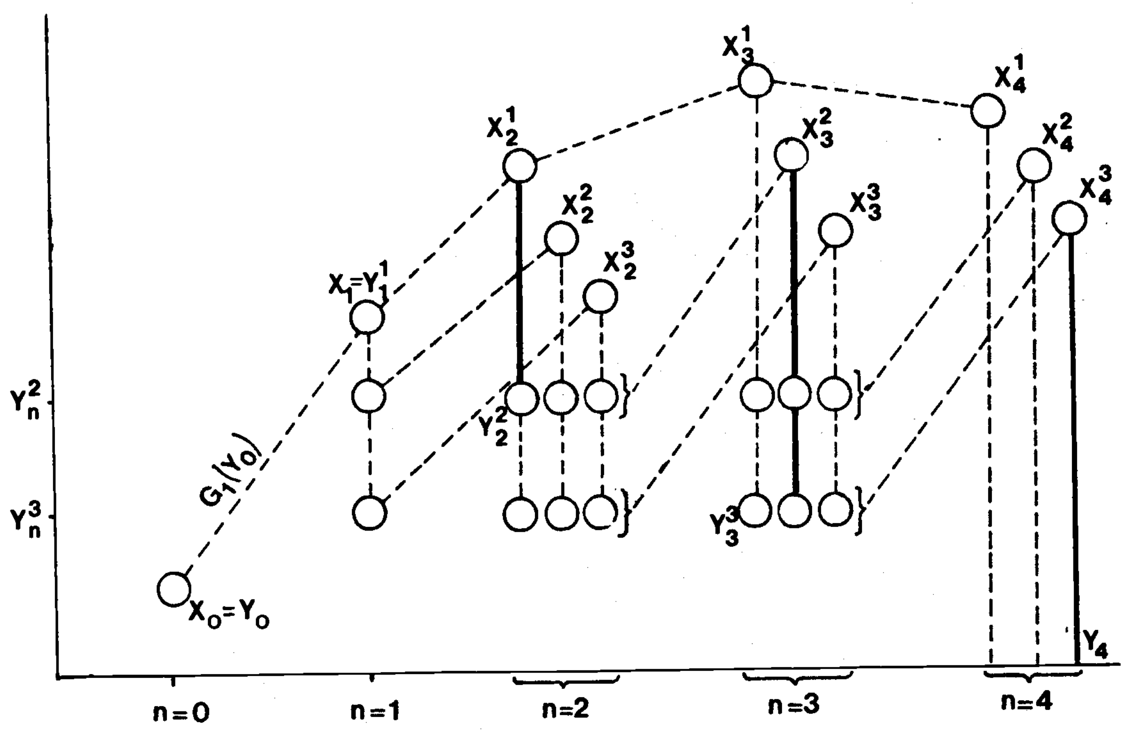

Figure 2. Network describing decision options for hypothetical stand: traditional DP algorithm. Full lines are arcs incorporated in the objective function, dotted lines are used in recursion.

The following volume equation is used to evaluate the term $[Y_n + G_{n+1}(Y_n)]$:

$$\log V_{t+5} = 1.5 + [(t+5)/t]*\log \sqrt{t}$$

where t represents age of the stand, and V is the standing volume. In figure 2 the node Y_o corresponds to the initial state ($X_o = Y_o = 500$ cu.ft.) of the stand at age 15; the sets $[Y_1]$, $[Y_2]$ and $[Y_3]$ are the feasible states at ages 20, 25 and 30 respectively; and the node Y_4 is the only feasible state at age 35. So the problem is to find the path yielding the maximum accumulated volume at rotation age.

The traditional DP method focuses the problem as identifying the best path for reaching each state at stage n; for example there are three different ways of obtaining a stocking level of 1500 cu.ft. at age 30 (i.e. reaching the node Y_3^2) one way is by thinning the natural stand, thus removing 4337-1500= 2837 cu.ft. at that age; the second option is by letting the stand grow from state Y_2^2 and removing 2545-1500=1045 cu.ft.; and the third way is by thinning the stand grown from Y_2^3 and then thinning a volume equal to 1995-1500=495 cu.ft. Thus in the traditional procedure the best option for reaching node Y_3^2 has to be selected as the one with the maximum cumulative volume removed so far. This implies also considering the volume removed up to the nodes from which Y_3^2 can be reached, $[Y_2^1, Y_2^2, Y_2^3]$, which by induction from the first stage are also the maximum for each node.

In the traditional DP algorithm the numerical expression of the recursive equation when solving, for instance, the state Y_2^2 becomes :

$$f_2(Y_2) = \max[0+2148;\ 492+1555;\ 992+871]$$
$$= \max[2148;\ 2047;\ 1863] = 2148$$

and this value is obtained by following the path $Y_o - Y_1^1 - Y_2^2$.

Since the problem has N=4 stages and K=3 states at each stage, except the last one when only one state is feasible, [K*(K-1) + 1]*(N-2) +2K =20 calculations (i.e. evaluations as above) are required to solve it. The maximum yield is obtained with the path $Y_o - Y_1^1 - Y_2^2 - Y_3^3 - Y_4$ and it is 5471 cu.ft.

To solve the problem with the new algorithm the network presented in Figure 3 is more appropriate, nevertheless the production data used are the same as that of Figure 2.

By following the algorithm presented in the previous section the first iteration is made to solve the set $[Y_1]$ which corresponds to age 20 (i.e. state 1). Since the problem is one of maximizing volume, equation

9 is appropriate but implemented as equation 10 for easy computation.

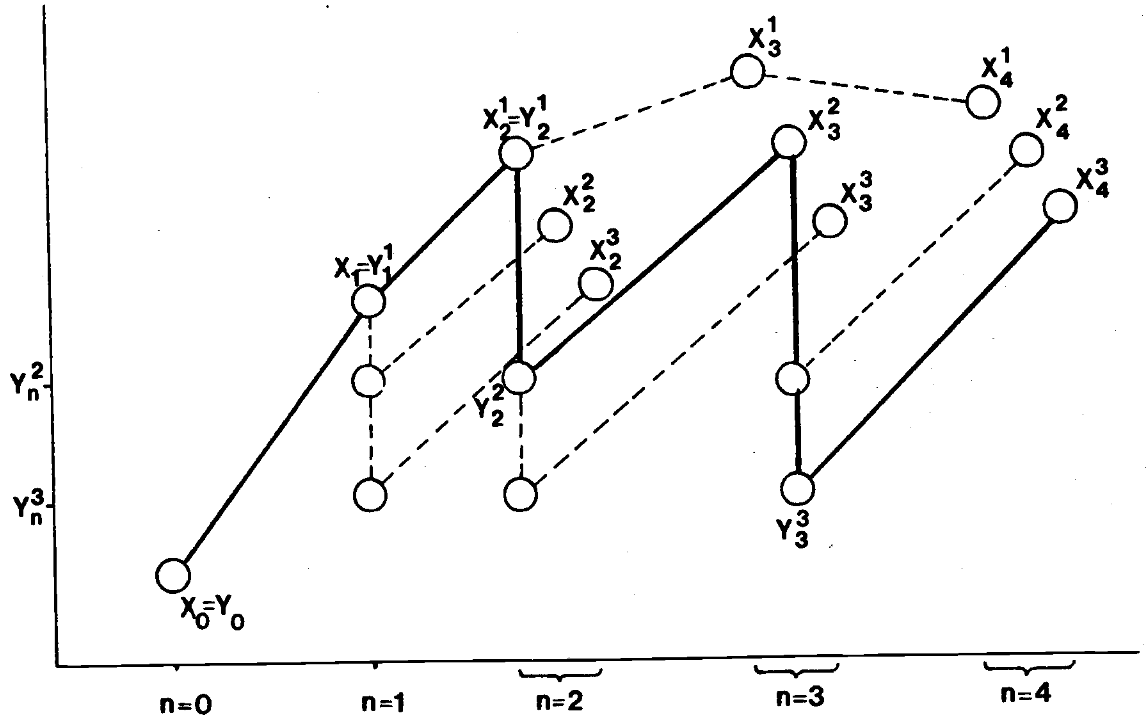

Figure 3. Network describing decision options for hypothetical stand: New algorithm. Full lines are arcs incorporated in the objective function, dotted lines are arcs used in recursion.

Therefore at age 20 the algorithm solves

$$f_1(Y_1) = \max \left\{ \begin{array}{l} [\ 0 + 3648] - 1992 + 1992 \\ [492 + 3055] - 1992 + 1992 \\ [992 + 2371] - 1992 + 1992 \end{array} \right\} = 3648$$

which corresponds to select node Y_1^1 as the nexus for all nodes in the next stage and all those nodes become permanently labeled with the value 3648 (cu.ft.). The next search is on the set $[Y_2]$ at age 25, then

$$f_2(Y_2) = \max \left\{ \begin{array}{l} [\ \ 0 + 4337] - 3648 + 3648 \\ [2148 + 2545] - 3648 + 3648 \\ [2648 + 1995] - 3648 + 3648 \end{array} \right\} = 4693$$

and this cumulative volume can be reached through the path $Y_o - Y_1^1 - Y^2{}_2$. The iteration for stage 3 implies the evaluation

$$f_3(Y_3) = \max \left\{ \begin{array}{l} [\ 0 + 3067] - 2545 + 4693 \\ [1045 + 2253] - 2545 + 4693 \\ [1545 + 1778] - 2545 + 4693 \end{array} \right\} = 5471$$

which is the optimum possible value to obtain from the stand and corresponds to selecting node Y_3^3 in the 3rd stage. Node Y_4 is the final stage so it is also solved by the previous iteration.

The optimum stocking regime is the same solution provided by traditional DP (path $Y_o - Y_1^1 - Y_2^2 - Y_3^3 - Y_4$); however the number of calculations was reduced to K*(N-1)=9 evaluations of the functional equation 12. In this case the correct evaluation of λ_n as the shadow price of the residual stand has lead to the optimal solution.

For small problems, as in this example, the savings in number of calculations are not so impressive, however the difference in computing effort between both algorithms is exponentially increased when the size of the problem is increased. A problem with 40 states and 12 stages will require 7690 calculations with traditional DP and 440 with the new method.

COMPARISON WITH AN EXISTING DP MODEL

Any comparison with an existing DP model requires the new algorithm to use identical production and cost functions. The growth model for Douglas fir embedded in DOPT corresponds to a version of DFIT (Bruce and others, 1977) with minor modifications introduced by Brodie and Kao (1979) when developing DOPT. These growth, cost and revenue functions were extracted from DOPT and utilized in the new algorithm, insuring a comparison over identical production surfaces. For the remainder of the paper we will refer to the traditional approach as the DOPT algorithm and the new approach as the PATH algorithm.

Both algorithms are written in FORTRAN 77 and were compiled with the same optimization level of the FORTRAN Compiler, NOS operating system for a Control Data Corp. CYBER 73/16 mainframe computer.

Two sample problems were stated and solved with each algorithm in order to illustrate the real differences in execution time. We include these examples to illustrate the practical advantage of the new approach and not merely as a software comparison.

The following data is shared by the two problems: the stand is a natural growth of Douglas-fir site index 140 (McArdle, 1961); it has not been precommercially thinned nor fertilized. The only decision variable is number of trees (in 15-trees intervals) left after thinning, and the state descriptors are age (in 10-year intervals), number of trees (in 15-tree intervals), and basal area (in multiples of 20 sq.ft./acre). The last two descriptors are required by DOPT. A $200 per acre regeneration cost is assumed to have been incurred before the age of the first thinning option (30 years).

Specifically each problem is defined through the following data:

Problem A:	-net interest rate	i= 4%
	-hauling costs	H= 50 \$/Mcu.ft.
Problem B:	-net interest rate	i= 2%
	-hauling costs	H= 100 \$/Mcu.ft.

where net interest rate refers to the real cost of capital, and logging and hauling costs are evaluated as described by Sessions (1979).

The feature used to demonstrate efficiency was the comparable computer times required to solve a given problem using DOPT and PATH. Comparisons should be between algorithms for the same problem and same computer. Table 1 presents the results in execution time for both, CYBER 73/16 mainframe computer and the IBM-XT 5160/087.

Table 1 Computing time on a mainframe and a personal computer.

	DOPT	PATH
a) CDC CYBER 73/16	[CPU secs]	
problem A	99.78	4.19
problem B	187.30	5.45
b) IBM-XT/DOS 3.10 [secs]1/		
problem A	4475	48
problem B	10775	57

1/ Timing with internal clock, includes processing and I/O operations. In DOPT reading and writing consumes a greater amount of time than processing.

As was expected the ratio in mainframe execution time between algorithms was about 30 times to solve this type of simple DP problem; for complex DP formulations such as those involving several decision variables and more state descriptors, the ratio will increase exponentially.

An interesting sidelight of the comparative analysis was that the solutions reported by the two algorithms present occasional discrepancies. It was expected that the solutions for a given problem would be identical since both methods are nested in a unique canonical form. Table 2 summarizes the optimum values for all rotations reported as solutions by each model.

Table 2 Soil expectation value for optimal stocking regimes, different rotation ages.

[$/acre]

	Problem A		Problem B	
Age	DOPT	PATH	DOPT	PATH
60	35.1	35.1	604.1	604.1
70	98.9	98.9	1097.9	1097.9
80	101.1	101.1	1305.0	1305.0
90	107.2(1)	108.6(1)	1379.8	1379.8
100	102.3	103.1	1393.4(1)	1386.4
110			1386.6	1405.9(1)
120				1396.4

(1) Soil expectation value for optimal rotation age.

For certain rotation lengths, PATH and DOPT tend to find identical optimal values of the objective function, and therefore the algorithms report the same optimum solution. However, there are instances where the algorithms report different solutions and consequently different objective function values. Since both algorithms utilize the same functions to grow the stand and to evaluate alternatives, these discrepancies have to be explained in terms of the characteristics of the solution methods here compared.

In the cases presented these slight discrepancies occur from an artifact of the "neighborhood storage" method of DOPT to reduce the number of feasible values of each state variable. In this method continuous stand variables are classified to discrete nodes and artifacts were noted by Kao (1980) when wider state spaces sometimes gave higher objective function values; alternatives are classified to different nodes and thus not eliminated from future stages. This discretization procedure, also known as the "coarse grid approach", has been long recognized as potential source of discrepancies in the optimal solution of dynamic programming problems (Nemhauser, 1966).

Additionally, a potential source of differences are the cost functions embedded in DOPT to evaluate the cost of logging thinned and clearcut volumes. The cost evaluation routine, developed by Sessions (1980), is composed by several equations forming an overall discontinuous or segmented cost function with independent variables being the parameters of the removed stand volume. These discontinuities may result in a source of inestability in the objective function values whenever the optimal solution includes, at any stage, a decision option being evaluated at any of the discontinuities. In the examples presented here the optimal solutions reported by both algorithms do not intersect the ill-conditioned poits of the cost function and therefore the solutions from either algorithm can be certified as optimal for each problem.

In general, it is expected to find such discrepancies when parallel mathematical methods are used to solve the same control problem. Discrepancies in optimal solutions were noted and analized by Everett (1963) when solving a problem with dynamic programming and LaGrange multipliers algorithms. Fan and Wang (1964, p78) also reported the same type of discrepancies when solving a replacement problem wth both a DP algorithm and a discrete Maximum Principle algorithm which, as pointed out earlier, uses the adjoint variables as an equivalent concept to the LaGrange multipliers.

Such discrepancies would only be noted in the case of parallel specifications as developed here. They can be controlled or eliminated by reducing the intervals and are reported here for additional insight into artifacts in DP solutions.

CONCLUSIONS AND EXTENSIONS

During the last decade dynamic programming and optimal control have shown their general suitability to solve the stand level control problem. The dynamic programming approach can provide numerical solutions to actual situations up to the point where the dimensionality of the problem being solved becomes a binding constraint. On the other hand, the optimal control approach (specifically the Maximum Principle) provides interesting economic insights to the nature of the problem by utilizing concepts such as shadow prices or opportunity cost of a given alternative, however its application has been restricted to the continuous-time version of the problem and simplified growth models.

Furthermore, as claimed by Nemhauser (1966), dynamic programming is an approach to optimization rather than a computational method for optimizing particular objective functions and specifications. An analyst may use any suitable method of optimization to construct and solve the recursive equation. The LaGrange multipliers of Everett (1963) provide such method to reformulate the traditional approach and simultaneously getting an interesting analogy with the discrete-time Maximum Principle.

It has also been demontrated that the increasing availability of combinatorial techniques during the last decades represent a source of useful optimization methods suitable to forestry applications that are otherwise intractable.

The method proposed in the previous sections has proved its feasibility with much less dimensionality and computation than traditional dynamic programming where some properties of the production, cost and revenue functions can be assumed. Comparison of the two methods using a modern stand growth simulator revealed a substantial increase in efficiency.

Growth models are becoming more prevalent in forestry applications and more complex in structure. The usefulness of optimization techniques is becomming widely recognized. Concerns about the dimensionality and computational burden of these newer models when transformed to an optimization framework, have been expressed, but the gains in efficiency demonstrated here, indicate a scope for further applications to increasingly complex problem structures.

The method permits utilization of additional computational refinements such as seeking methods when the response surface satisfies requirements of concavity or unimodality.

Nevertheless, we recognize that further investigation of the estimation of the LaGrange multipliers is required. The opportunity cost of the residual stand utilized in the example (equal to the stumpage value of the stand after one more growth period) is not necessarily a rule but a good initial guess for other applications of the method. Even though Everett (1963) does not impose restrictions on the functions being optimized (since his method implies a search for the multiplier values), it appears reasonable to suggest, for stand management problems, a preliminary investigation of the production surface (i.e., the stand growth simulator) being utilized. Conditions of quasiconcavity in the growth function, and continuity or at least monotonicity in the revenue and cost functions are desirable if the search in the value for the multipliers is to be avoided or minimized. The time spent in this investigation and in formulating the problem can save a vast amount of computing time. Judgement and preliminary investigation can substitute for the full enumeration of a network used in the traditional dynamic programming approach.

BIBLIOGRAPHY

AMIDON, E. L., and G. S. AKIN. 1968. Dynamic programming to determine optimum levels of growing stock. Forest Sci 14:287-291.

ARIMIZU, T. 1958. Regulation of the cut by dynamic programming. J Operations Res Soc Japan. 1(4):175-182.

BAUMOL, W. J. 1965. Economic theory and operations analysis. 2nd Ed. Prentice-Hall, Inc, NJ. 606 p.

BELLMAN, R. 1957. Dynamic programming. Princeton University Press, Princeton, New Jersey, 340 p.

BELLMAN, R. and S. E. DREYFUS. 1962. Applied dynamic programming. Princeton University Press. Princeton, New Jersey, 363 p.

BRODIE, J. D. and C. KAO. 1979. Optimizing thinning in Douglas-fir with three-descriptor dynamic programming to account for accelerated diameter growth. Forest Sci 25:665-672.

BRODIE, J. D. and R. G. HAIGHT. 1985. Optimization of silvicultural investment for several types of stand projection systems. Can J For Res 15:188-191.

BRUCE, D., D. J. DEMARS, and D. L. REUKEMA. 1977. Douglas-fir managed yield simulator and DFIT: user's guide. USDA Forest Service. Gen Tech Rep PNW-57.

CAWRSE, D.C., D.R. BETTERS, and B.M. KENT. 1984. A variational technique for determining optimal thinning and rotation schedules. Forest Sci 30: 793-802.

CHANG, S. J. 1984. A simple production function model for variable density growth and yield modeling. Can J For Res 14:783-788

CLARK, C.W. 1976. Mathematical bioeconomics. John Wiley & Sons, NY 352p.

CLUTTER, J. L. 1963. Compatible growth and yield models for loblolly pine. Forest Sci 9:354-371.

DENARDO, E. J. 1982. Dynamic programming: models and applications. Prentice-Hall, Inc, 227 p.

DIJKSTRA, E. W. 1959. A note on two problems in connexion with graphs. Numerische Mathematik 1:269-271.

DONNELLY, D.M. and D.R. BETTERS. 1985. Optimal control for even-aged stand harvest scheduling. Paper presented at 1985 Symp on Syst Analysis in Forest Res, SAF, Athens Ga. 25p.

DORFMAN, R. 1969. An economic interpretation of optimal control theory. Am Economic Rev 59:817-831.

DREYFUS, S. E. 1957. Computational aspects of dynamic programming. Operations Research 5:409-415.

DYKSTRA, D. P. 1984. Mathematical programming for natural resource management. McGraw-Hill, Inc, 318 p.

EVERETT, H. 1963. Generalized LaGrange multiplier method for solving problems of optimum allocation of resources. Operations Research 11:399-417.

FAN, L. and C. WANG. 1964. The Discrete Maximum Principle. John Wiley and Sons, Inc. NY. 158 p.

FOX, B. L. 1966. Discrete optimization via marginal analysis. Management Sci 13:210-215.

FOX, B. L. and D. M. LANDI. 1970. Searching for the multiplier in one-constraint optimization problems. Operations Research 18:253-262.

HAIGHT, R. G., J. D. BRODIE, and W. G. DAHMS. 1985. A dynamic programming algorithm for optimization of lodgepole pine management. Forest Sci 31:321-330.

HANN, D. W. and J. D. BRODIE. 1980. Even-aged management: basic managerial questions and available or potential techniques for answering them. USDA Forest Serv Gen Tech Rep INT-83.

HAX, A. C. 1973. Aggregate capacity planning, a review. M.I.T. Operations Res Center, Cambridge. Tech Rep 85, 46 p.

HILLIER, F. S. and G. J. LIEBERMAN. 1980. Introduction to Operations Research. 3rd. edition. Holden-Day, Inc, 829 p.

HU, T. C. 1982. Combinatorial algorithms. Addison-Wesley Pub Co, 292 p.

INTRILIGATOR, M. D. 1971. Mathematical optimization and economic theory. Englewood Cliffs, Prentice Hall, New Jersey 448p.

JOHNSON, K. N. and K. E. SLEAVIN. 1984. DP-DFSIM overview and user's guide. USDA Forest Service, Fort Collins, Colorado 46 p.

KAO, C. 1979. A study of optimal timing and intensity of silvicultural practices - commercial and precommercial thinning, fertilization and regeneration effort. Ph. D. thesis. Oregon State University, Corvallis, Oregon 219 p.

McDONOUGH, J.M. and P.E. PARK. 1975. A discrete maximum principle solution to an optimal control formulation of timberland management problems. Research Paper PROSE Inc., Intersc Publ NY 77p.

MARTIN, G. L. and A. R. EK. 1981. A dynamic programming analysis of silvicultural alternatives for red pine plantations in Wisconsin. Can J Forest Res 11:370-379.

MCARDLE, R. E., W. H. MEYER, and D. BRUCE. 1961. The yield of Douglas-fir in the Pacific Northwest. USDA Forest Service Tech Bull 201, 74 p.

NASLUND, B. 1969. Optimal rotation and thinning. Forest Sci 15:446-451.

NEMHAUSER, L. G. 1966. Introduction to dynamic programming. J. Wiley and Sons, Inc 256 p.

PAREDES G. and J. D. BRODIE. 1985. Efficient computation considerations in optimization of stand-level management regimes. Paper presented at 1985 Symp on Syst Analysis in Forest Res., SAF, Athens, Ga. 11p.

RIITTERS, K., J. D. BRODIE, and D. W. HANN. 1982. Dynamic programming for optimizing of timber production and grazing in ponderosa pine. Forest Sci 28:517-526.

SCHREUDER, G. F. 1971. The simultaneous determination of optimal thinning schedule and rotation for an even-aged forest. Forest Sci 17:333-339.

SESSIONS, J. 1979. Effects of harvesting technology upon optimal stocking regimes of forest stands in mountainous terrain. PhD thesis, School of Forestry, Oreg State Univ, Corvallis. 259 p. (Univ microfilms, Ann Arbor 79-11913).

SETHI, S. D. 1973. An application of optimal control theory in forest management. J Management Sci and Appl Cybern Vol 2 No 1, India.

VALSTA, L. T., and J. D. BRODIE. 1985. An economic analysis of hardwood treatment in loblolly pine plantations: a whole rotation dynamic programming approach. Society of American F

APPENDIX

APPENDIX   Program PATH

```
      PROGRAM PATH
C***************************************************************
C              PROGRAM PATH. FORTRAN 5 INTERACTIVE VERSION AS      *
C              CORRECTED ON AUGUST 1986.                           *
C              THIS IS AN ALGORITHM TO OPTIMIZE  ROTATION AND      *
C              THINNING REGIME IN EVEN-AGED STANDS OF DOUGLAS      *
C              FIR. IT IS BASED ON THE GROWTH  MODEL DFIT  BY      *
C              BRUCE ET. AL. (1977)  AS  MODIFIED IN  DOPT BY      *
C              BRODIE AND KAO (1979).                              *
C***************************************************************
C
C              SEE ABOVE REFERENCES FOR A DEFINITION OF GROWTH
C              VARIABLES AND STAND PARAMETERS
C
      DIMENSION THVAL(39),THVLM(39),THRUET(39),THRUEBA(39),FIVAL(39)
      DIMENSION OPTRUET(20),OPTVOL(20),OPTBA(20),HRVOL(20)
      DIMENSION VCUT(39),DMEAN(39),OPOLVR(39),OPOLHR(39)
      DIMENSION HRVSTD(14),HRVSTB(14),HRVSTP(14),HRVSTS(14),HRVSTC(14)
      DIMENSION NN(15),TNORM(14),GNORM(14),VNORM(14),Z(14),DRATIO(5)
      DIMENSION OUT(14,6),NITMP(14),IPAGE(14),XAGE(14)
      DIMENSION JFERT(14),DATIOD(14)
      COMMON TAGE,TBASE,SITE,TRATIO,GRATIO,VGRATIO
      COMMON PMORTN(25),PMORTG(25),PMORTV(25),YMORTN(250),YMORTG(250)
      COMMON TNONMER(25),GNONMER(25),VNONMER(25)
      INTEGER OPOLVR,OPOLHR,W
      CHARACTER *10 SKIP(14)
       OPEN(6, FILE= 'OUTPUT')
       OPEN(5, FILE='INPUT')
       DRATIO(1)=1.0
       TRATIO=1.0
       GRATIO=1.0
      N=1
      L=1
      MAXVR=39
C-------------------------------------------------------------
C           DATA INPUT, AS IN DOPT
C-------------------------------------------------------------
      WRITE(*,6000)
 6000 FORMAT(//'           PROGRAM  P A T H'//
     1  ' A ROTATION AND THINNING REGIME OPTIMIZER FOR EVEN-AGED '/
     2  ' STANDS OF DOUGLAS-FIR. '/
     3  ' IT USES DFIT AS A GROWTH MODEL IN A RECURSION PROCEDURE '/
     4  ' THAT EVALUATES DECISION OPTIONS BY PROJECTING THE RESIDUAL '/
     5  ' STAND TO THE NEXT STAGE (10-YEARS LOOKAHEAD) '/
     6  ' END ALL ENTRIES WITH RETURN.'/
     7  ' ZEROES MUST BE ENTERED FOR VALUES NOT USED.'/)
 7001 WRITE(*,6001)
 6001 FORMAT(' ENTER TREE THINNING INTERVAL (USE 15)')
      ASSIGN 7001 TO W
      READ (*,*,END=7999) INTVR
 7002 WRITE(*,6002)
```

```
6002 FORMAT(' ENTER MCCARDLE 100 YEAR SITE INDEX')
     ASSIGN 7002 TO W
     READ (*,*,END=7999) SITE
     IF (SITE .LT. 10 .OR. SITE .GT. 300) GO TO 8999
     TESTA=13.22-0.033*SITE
7003 WRITE(*,6003)
6003 FORMAT(' ENTER 0 FOR NORMAL STAND OR 1 FOR NON-NORMAL STAND')
     ASSIGN 7003 TO W
     READ (*,*,END=7999) TESTN
     IF (TESTN .NE. 0 .AND. TESTN .NE. 1) GO TO 8999
7004 WRITE(*,6004)
6004 FORMAT(' ENTER AGE OF FIRST COMMERCIAL ENTRY FOR NORMAL STAND,'/
    1  '     (NORMALLY 30 OR GREATER)'/
    2  '     OR CURRENT AGE FOR NON-NORMAL STAND')
     ASSIGN 7004 TO W
     READ (*,*,END=7999) TBASE
     IF (TESTN .EQ. 0 .OR. TBASE .GT. TESTA) GO TO 7005
     WRITE (*,6023) TESTA
6023 FORMAT(' TRY AGAIN.'/' ENTER A VALUE GREATER THAN ',F3.0)
     GO TO 7004
7005 WRITE(*,6005)
6005 FORMAT(' ENTER 0 FOR NO PRECOMMERCIAL THINNING OR'/
    1  '     OR 1 FOR PRECOMMERCIAL THINNING (PCT)')
     ASSIGN 7005 TO W
     READ (*,*,END=7999) TESTP
     IF (TESTP .NE. 0 .AND. TESTP .NE. 1) GO TO 8999
7006 WRITE(*,6006)
6006 FORMAT(' ENTER 0 FOR NO FERTILIZATION OR 1 FOR FERTILIZATION')
     ASSIGN 7006 TO W
     READ (*,*,END=7999) TESTF
     IF (TESTF .NE. 0 .AND. TESTF .NE. 1) GO TO 8999
7008 WRITE(*,6008)
6008 FORMAT(' ENTER DECIMAL RATE OF INTEREST (FOR 3%, ENTER .03)'/
    1  '     AND DECIMAL RATE OF REAL PRICE INCREASE'/
    2  '     (USUALLY BETWEEN .01 AND .03)')
     ASSIGN 7008 TO W
     READ (*,*,END=7999) R, R1
7010 WRITE(*,6010)
6010 FORMAT(' ENTER THE HIGHEST NUMBER OF TREES AFTER PCT FOR A',
    1  ' NORMAL STAND'/
    2  '     OR THE TOTAL NUMBER OF TREES IN A NON-NORMAL STAND')
     ASSIGN 7010 TO W
     READ (*,*,END=7999) STREE
7011 WRITE(*,6011)
6011 FORMAT(' ENTER BASAL AREA FOR A NON-NORMAL STAND',
    1  ' OR 0 FOR A NORMAL STAND')
     ASSIGN 7011 TO W
     READ (*,*,END=7999) SBA
7014 WRITE(*,6014)
6014 FORMAT(' ENTER REGENERATION COST PER ACRE'/
    1  '     AND HAUL COST IN $ PER THOUSAND CUBIC FEET')
     ASSIGN 7014 TO W
     READ (*,*,END=7999) REGENC, HAUL
1001 WRITE(*,6018)
     S=1.0
```

```
      NDRATIO=1
      DRATIO(1)=1.0
      NADD=0
 6018 FORMAT(' YOUR DATA INPUT HAS BEEN ACCEPTED  '/
     1  '            SO I AM RUNNING              ')
      GO TO 7022
 7999 WRITE(*,6999)
 6999 FORMAT(' ZEROES MUST BE ENTERED FOR VALUES NOT USED.'/
     1   ' TRY AGAIN.')
      GO TO W,(7001,7002,7003,7004,7005,7006,7008,7010,7011,7014)
 8999 WRITE(*,9999)
 9999 FORMAT(' SORRY, YOUR DATA IS NOT ACCEPTABLE.  TRY AGAIN.')
      GO TO W,(7003,7004,7005,7006)
      CLOSE (UNIT=5)
C
 7022 WRITE(*,6020)
 6020 FORMAT('1'///' P A T H - DEVELOPED AT OREGON STATE UNIVERSITY',
     1   ' BY GONZALO'/'         AND DOCUMENTED IN 1985 SUMMER.'/
     2   '           FOR QUESTIONS, CONTACT D. BRODIE AT FOREST '/
     3   '           MANAGEMENT DEPARTMENT, PHONE 503-754-2796'/
     4   '           OR 503-754-4951.'//)
C-----------------------------------------------------------------
C             ECHO DATA TO OUTPUT FILE
C-----------------------------------------------------------------
      WRITE(6,3) TBASE,SITE,R,R1,REGENC,S,HAUL
3     FORMAT(' AGE IS',F6.1//' SITE INDEX IS',F7.1//' INTEREST RATE IS',
     $F5.3//' PRICE INFLATION RATE IS',F6.3//' REGENERATION COST',
     $' IS $',F7.2//' LOGGING COST SHIFT FACTOR IS',F6.2//
     $' HAUL COST PER THOUSAND CUBIC FEET IS $',F7.2)
      WRITE(6,13)(DRATIO(I),I=1,NDRATIO)
 13   FORMAT(/' THINNING RATIOS(DCUT/DTOTAL) ARE:',5F6.2)
      SITEOLD=SITE
      TAGE=TBASE
C-----------------------------------------------------------------
C             EVALUATES PARAMETERS  NORMAL STAND
C             CALCULATE NATURAL STAND AT BASE AGE
C-----------------------------------------------------------------
C
      BHAGE=TAGE-13.22+0.033*SITE
      DNATURE=10**(0.1097-3.4857/BHAGE**0.25+1.0531*ALOG10(SITE))
      TNATURE=10**(3.9108+5.2306/BHAGE**0.25-1.5803*ALOG10(SITE))
      GNATURE=10**(1.8669-1.7408/BHAGE**0.25+0.5259*ALOG10(SITE))
      HT=10**(0.1567-15.673/TAGE+ALOG10(SITE))
      VG=10**(-0.0282+0.7917*ALOG10(HT))
      VGRATIO=VG
      VNATURE=GNATURE*VGRATIO
      TRATIO=STREE/TNATURE
      GRATIO=SBA/GNATURE
      IF(TESTN.EQ.0.) TRATIO=1.
      IF(TESTN.EQ.0.) GRATIO=1.
C-----------------------------------------------------------------
C             CALCULATE MERCHANTABLE PART AND MORTALITY LATER PART
C-----------------------------------------------------------------
      CALL SUBMORT(DNATURE)
      L=TBASE/10.-2.
      IF(L.LT.1) L=1
      GMERCH=GNATURE-GNONMER(L)/GRATIO
      TMERCH=TNATURE-TNONMER(L)/TRATIO
      VMERCH=VNATURE-VNONMER(L)/GRATIO
C--------------------------------------------------------------------
C             DEFINES INDEX FOR INITIAL STATE SETTING I MAXIM
C--------------------------------------------------------------------
16    I=TMERCH*TRATIO/INTVR+1.999999
      IF(I.LT.1) I=1
      IF(I.GT.MAXVR) I=MAXVR
      TRUET=TMERCH*TRATIO
      TRUEBA=GMERCH*GRATIO
      VLM=VMERCH*GRATIO
      VAL=-REGENC
C--------------------------------------------------------------------
C             WITH NS THE THINNINGS ARE RESTRICTED TO A 50%
C             IN THE FIRST STAGE
C--------------------------------------------------------------------
17    NS=TMERCH/(2.*INTVR)+1.999999
      IF((NS-1.)*INTVR.GT.TRUET) NS=TRUET/INTVR+1.999999
      IF(NS.LT.1) NS=1
      IF(NS.GT.MAXVR) NS=MAXVR
C--------------------------------------------------------------------
C             CALCULATE NATURAL STAND AND MERCHANTABLE TREES.
C--------------------------------------------------------------------
      DO 18 I=L,14
      AB=30+10*(I-1)
      B=AB-13.22+0.033*SITE
      TNORM(I)=10**(3.9108+5.2306/B**0.25-1.5803*ALOG10(SITE))
      GNORM(I)=10**(1.8669-1.7408/B**0.25+.5259*ALOG10(SITE))
      VNORM(I)=10**(1.9628-12.4083/AB-1.7408/B**.25+1.3176*ALOG10(SITE))
18    Z(I)=TNORM(I)-TNONMER(I)/TRATIO
C--------------------------------------------------------------------
C             CREATE PARAMETERS TO ITERATE
C--------------------------------------------------------------------
      SITE50=21.5-0.18127*30.+0.72114*SITE
      TMORT1=0.0
      GMORT1=0.0
      VMORT1=0.0
      GMORT2=0.0
      VMORT2=0.0
      VFERT=0.0
C
      OPVALUE=-99999.99
C--------------------------------------------------------------------
C             STARTS THE LOOP FOR EACH STAGE
C--------------------------------------------------------------------
 19     CONTINUE
         WRITE(6,802)N
 802     FORMAT(//' STAGE NUMBER ',I2//)
       TAGE=TBASE+10.*(N-1)
      IF(N.EQ.1) GOTO 25
      NS=2
      HT=10**(0.1567-15.673/(TAGE-10.)+ALOG10(SITE))
      VG=10**(-0.0282+0.7917*ALOG10(HT))
```

```
25    KJ=TAGE/10.
      IF(KJ.LT.1) KJ=1
      IF(KJ.GT.14) KJ=14
C------------------------------------------------------------------------
C             REPLACES NUMBER OF TREES AND VOLUME BY OPTIMUM NODE
C------------------------------------------------------------------------
        IF(N.EQ.1) GO TO 300
        VLM=OPTVOL(N-1)
       TRUET=OPTRUET(N-1)
 300  BHAGE=(TAGE-10.)-13.22+0.033*SITEOLD
      GBOUND=0.
      IF(N.EQ.1.OR.TAGE.GT.70..OR.TESTF.EQ.0.) IQUANT=1
C------------------------------------------------------------------------
C             NEXT OUTPUT STATEMENTS ARE OPTIONAL FOR
C             DEBUGING PURPOSES
C------------------------------------------------------------------------
C
C(DEBG OPT 1)         WRITE(6,777)N,L,TRUET,OPTBA(N-1),VLM
C(DEGB OPT 1)  777    FORMAT(' STAGE N=',I2,' L=',I2/' TRUET =',F6.1/
C(DEBG OPT 1)       $ ' TRUEBA=',F7.1/' VLM=',F7.1/)
C
C------------------------------------------------------------------------
C             STARTS THE LOOP FOR THE I THINNING OPTIONS ACCORDING
C             TO STANDING NUMBER OF TREES
C
C             IN THE FIRST STAGE THIN FIRST THEN GROW
C------------------------------------------------------------------------
      IF(N.EQ.1) GOTO 28
      VOL=VLM+VNONMER(N+L-1)
C------------------------------------------------------------------------
C             IF IT IS NOT THE FIRST STAGE CALL GROWTH
C------------------------------------------------------------------------
      CALL GROWTH(TRUET,VOL,VMER,VMORT1,GMER,GMORT1,N,TESTP)
C
      VLM=VMER
      TRUEBA=GMER
      PNORM=TRUET/Z(N+L-2)
      TMORT1=(Z(N+L-2)-Z(N+L-1))*PNORM
      IF(TESTP.EQ.1.) TMORT1=0.
      TRUET=TRUET-TMORT1
C------------------------------------------------------------------------
C             LOOP FERTILIZATION ELIMINATED IN THIS VERSION OF PATH
C------------------------------------------------------------------------
28    NT=TRUET/INTVR+1.999999
      IF(NT.LT.1) NT=1
      IF(NT.GT.MAXVR) NT=MAXVR
C
      DIAM=SQRT(TRUEBA/(0.005454154*TRUET))
      VOLMOR=VMORT1
      BALMOR=GMORT1
      DO 41 K=NS,NT
 41   THVAL(K)=0.0
      FIVAL(K)=0.0
      THRUET(K)=0.0
      THVLM(K)=0.0
```

```
C-----------------------------------------------------------------
C              NS IS CONSTRAINED TO 50% STOCKING
C-----------------------------------------------------------------
      DO 40 K=NS,NT
      TAGE=TBASE+10.*(N-1)
C-----------------------------------------------------------------
C              FOR K=NT WHICH IS NO THINNING
C-----------------------------------------------------------------
      IF(K.LT.NT) GOTO 32
      TMPBA=TRUEBA
      TMPVLM=VLM
      CUT=0.
      REV=0.0
      GOTO 35
C-----------------------------------------------------------------
C              NOW FOR K<NT WHICH ARE THE THINNINGS
C-----------------------------------------------------------------
32    TMPBA=TRUEBA*(1.-DRATIO(1)**2*(TRUET-(K-1)*
     $INTVR)/TRUET)
      IF(K.EQ.1) TMPBA=0.
      IF(TMPBA.LT.0.) GOTO 40
      TMPVLM=VLM*TMPBA/TRUEBA
      CUT=VLM-TMPVLM+VOLMOR
      DIAM1=SQRT((TRUEBA-TMPBA+BALMOR)/(.005454154*(TRUET-
     $(K-1)*INTVR+TMORT1)))
C-----------------------------------------------------------------
C              CALCULATE CUBIC VOL TO A 4-INCH TOP
C-----------------------------------------------------------------
      HD=10**(.1567-15.673/TAGE+ALOG10(SITE))
      N1=(TAGE-20.)/10.
      ALLTREE=TRUET+TNONMER(N1)
      HM=HD*(3040.-ALLTREE)/3000.
      IF(HM.GT.HD) HM=HD
      VOL4=CUT*(.8758+.001049*HM-.000002824*HM**2+.3221/DIAM1-
     $45.647/DIAM1**3)
      VAR=4.0725-0.065722*SITE+0.00001508*TAGE*SITE**2
      TRUED=SQRT(DIAM1**2-VAR)
      IF(TRUED.GT.22.)TRUED=22.
C-----------------------------------------------------------------
C              EVALUATES PNW FOR THINNED VOLUMES
C-----------------------------------------------------------------
      REV=REVNOW(VOL4,TRUED,S,R1,TAGE,HAUL)
C
35    FERTCST=0
      TMPVAL=(REV-(FERTCST)*(1+R)**10)/((1.+R)**TAGE)+VAL
      IF(K.GE.2) DIAM=SQRT(TMPBA/(0.005454154*(K-1)*INTVR))
      DMEAN(K)=DIAM
      THVAL(K)=TMPVAL
      OPOLVR(K)=(I-1)*INTVR
      THRUEBA(K)=TMPBA
      IF(TMPBA.GT.GBOUND) GBOUND=TMPBA
      THRUET(K)=(K-1)*INTVR
      IF(K.EQ.1) THRUET(K)=0.
      IF(K.EQ.NT) THRUET(K)=TRUET
      IF(K.EQ.1) HRVSTB(KJ)=TRUEBA
      THVLM(K)=TMPVLM
      VCUT(K)=CUT
C-----------------------------------------------------------------------
C              IN (K) ARE STORED THE PARAMETERS OF THE STAND
C              CORREPONDING  TO THE STATE BEING EVALUATED.
C              NOW ITERATES TO PROJECT THE RESIDUAL STAND.
C              A NUMBER OF IX LOOKAHEAD PERIODS CAN BE USED
C-----------------------------------------------------------------------
        IPPHH=IPLHOR/10-2
        IX=1
        XTRUET=THRUET(K)
        XVLM=THVLM(K)
         XTAGE=TAGE
C
        DO 800 IIX=1,IX
C-----------------------------------------------------------------------
C              NEXT FORMAT STATEMETS (779 AND 776) FOR
C              DEBUGING PURPOSES
C-----------------------------------------------------------------------
C
C(DEBG OPT 2)         WRITE(6,779)IIX,IX,XVLM,XTRUET
C(DEBG OPT 2) 779  FORMAT(' IIX=',I3,' IX=',I3/' XVLM ',F6.1/' XTRUET ',F6.1)
C
        NX=N+IIX
        XVOL=XVLM + VNONMER(NX+L-1)
        TAGE=XTAGE+10.*IIX
C
C(DEBG OPT 3)      WRITE(6,776)XVOL,VNONMER(NX+L-1),TAGE
C(DEBG OPT 3) 776  FORMAT(' XVOL',F6.1/' VNONMER ',F6.1/' TAGE ',F6.1)
C
        CALL GROWTH(XTRUET,XVOL,VMER,VMORT1,GMER,GMORT1,NX,TESTP)
C
        XVLM=VMER
        XPNORM=XTRUET/Z(NX+L-2)
        XMORT1=(Z(NX+L-2)-Z(NX+L-1))*XPNORM
        IF(TESTP.EQ.1.)XMORT1=0.
        XTRUET=XTRUET-XMORT1
        IF(IIX.LT.IX) GO TO 800
C
        XCUT=XVLM+VMORT1
        XDIAM1=SQRT((GMER+GMORT1)/(0.005454154*(XTRUET+XMORT1)))
      XMM=15.673/TAGE
         HD=10**(.1567-XMM+ALOG10(SITE))
        N1=(TAGE-20.)/10.
        ALLTREE=XTRUET+TNONMER(N1)
        HM=HD*(3040.-ALLTREE)/3000.
        IF(HM.GT.HD)HM=HD
        VOL4=XCUT*(0.8758+0.001049*HM-0.000002824*HM**2+0.3221/XDIAM1-
     $45.647/XDIAM1**3)
      VAR=4.0725-0.065722*SITE+0.00001508*TAGE*SITE**2
        TRUED=SQRT(XDIAM1**2-VAR)
        IF(TRUED.GT.22.)TRUED=22.
C-----------------------------------------------------------------------
C              EVALUATES PNW OF RESIDUAL STAND
C-----------------------------------------------------------------------
```

```
C
C
       REV=REVNOW(VOL4,TRUED,S,R1,TAGE,HAUL)
C
       CORTA=REV/((1.+R)**TAGE)
       FIVAL(K)=CORTA+TMPVAL
C
 800   CONTINUE
C
       IF(FIVAL(K).LT.OPVALUE) GO TO 850
       OPVALUE=FIVAL(K)
       PREVAL=THVAL(K)
       OPTRUET(N)=THRUET(K)
       OPTVOL(N)=TMPVLM
       OPTBA(N)=THRUEBA(K)
       STMPAGE=CORTA
       HRVOL(N)=VCUT(K)
C
 850   CONTINUE
       WRITE(6,778)K,NT,FIVAL(K)
 778   FORMAT(' STATE PARAMETERS ',2(I3,2X),'FIVAL ',F7.1)
  40   CONTINUE
C-----------------------------------------------------------------------
C              AGE AND STAGE INDICATORS FOR NEXT ITERATION
C-----------------------------------------------------------------------
        N=N+1
        TAGE=XTAGE+10.
        VAL=PREVAL
C-----------------------------------------------------------------------
C
C               WRITE FOR EACH STAGE THE OPTIMUM SUBPATH
C-----------------------------------------------------------------------
       WRITE(6,77)XTAGE
 77    FORMAT(' OPTIMUM NODE AT STAGE  ',F4.0,' YEARS'//)
       WRITE(6,78)OPTRUET(N-1)
 78    FORMAT('RESIDUAL NUMBER OF TREES',F6.1/)
       WRITE(6,79)OPTVOL(N-1)
 79    FORMAT(' RESIDUAL VOLUME          ',F7.1/)
       WRITE(6,82)HRVOL(N-1)
 82    FORMAT(' THINNED VOLUME           ',F7.1/)
       WRITE(6,81)PREVAL,STMPAGE
 81    FORMAT(' PREVIOUS VAL=',F7.1,' STUMPAGE AGE 100 ',F7.1/)
       WRITE(6,80)OPVALUE
 80    FORMAT(' ACCUMULATED PNW          ',F7.1////)
       IF(N.EQ.IPPHH)GO TO 955
       GO TO 19
955   CONTINUE
      CLOSE(6)
      END
C***********************************************************************
C             SUBROUTINE SUBMORT(DD)                                   *
C***********************************************************************
      SUBROUTINE SUBMORT(DD)
      COMMON TAGE,TBASE,SITE,TRATIO,GRATIO,VGRATIO
      COMMON PMORTN(25),PMORTG(25),PMORTV(25),YMORTN(500),YMORTG(500)
      COMMON TNONMER(25),GNONMER(25),VNONMER(25)
C-----------------------------------------------------------------------
C             CALCULATE PERIODIC MORTALITY
C             STARTS FROM THE AGE OF FIRST THINNING
C-----------------------------------------------------------------------
      B30=30.-13.22+0.033*SITE
      D30=10**(0.1097-3.4857/B30**0.25+1.0531*ALOG10(SITE))
      DM=(D30/0.875)*0.75
      DL=0.698*D30
      L=TBASE/10.-2.
      IF(L.LT.1) L=1
      TNONMER(L)=10**(3.8622+3.1994*ALOG10(DM)-4.7*ALOG10(DD))*TRATIO
      YMORTN(L)=TNONMER(L)
      GNONMER(L)=10**(1.4034+4.9394*ALOG10(DM)-4.44*ALOG10(DD))*GRATIO
      YMORTG(L)=GNONMER(L)
      VNONMER(L)=GNONMER(L)*VGRATIO*TARIF(DD)/TARIF(DL)
      DO 10 JJ=L,18
      AGE=30+JJ*10.
      BHAGE=AGE-13.22+0.033*SITE
      D=10**(0.1097-3.4857/BHAGE**0.25+1.0531*ALOG10(SITE))
      TNONMER(JJ+1)=10**(3.8622+3.1994*ALOG10(DM)-4.7*ALOG10(D))*TRATIO
      GNONMER(JJ+1)=10**(1.4034+4.9394*ALOG10(DM)-4.44*ALOG10(D))*GRATIO
      DMORTL=SQRT((GNONMER(JJ+1)/TNONMER(JJ+1))/0.005454154)
      HT=10**(0.1567-15.673/AGE+ALOG10(SITE))
      VG=10**(-0.0282+0.7917*ALOG10(HT))
      TAVE=(VG*TARIF(D)+VGRATIO*TARIF(DD))/2.
      VNONMER(JJ+1)=TAVE/TARIF(DMORTL)*GNONMER(JJ+1)
      PMORTN(JJ)=(TNONMER(JJ)-TNONMER(JJ+1))
      PMORTG(JJ)=(GNONMER(JJ)-GNONMER(JJ+1))
      DMORT=SQRT((PMORTG(JJ)/PMORTN(JJ))/0.005454154)
      PMORTV(JJ)=TAVE/TARIF(DMORT)*PMORTG(JJ)
10    CONTINUE
C-----------------------------------------------------------------------
C             CALCULATE YEARLY MORTALITY
C             STARTS FROM THE AGE OF FIRST THINNING
C-----------------------------------------------------------------------
      LL=(L-1)*10+1
      DO 20 II=LL,180
      AGE=30.+II
      BHAGE=AGE-13.22+0.033*SITE
      D=10**(0.1097-3.4857/BHAGE**0.25+1.0531*ALOG10(SITE))
      YMORTN(II+1)=10**(3.8622+3.1994*ALOG10(DM)-4.7*ALOG10(D))
      YMORTG(II+1)=10**(1.4034+4.9394*ALOG10(DM)-4.44*ALOG10(D))
      YMORTN(II)=(YMORTN(II)-YMORTN(II+1))*TRATIO
      YMORTG(II)=(YMORTG(II)-YMORTG(II+1))*GRATIO
20    CONTINUE
      RETURN
      END
C
C***************************************************************
C             SUBROUTINE GROWTH                                *
C***************************************************************
      SUBROUTINE GROWTH(TREE,V,VMER,VMORT1,GMER,GMORT1,M,TESTP)
      COMMON TAGE,TBASE,SITE,TRATIO,GRATIO,VGRATIO
      COMMON PMORTN(25),PMORTG(25),PMORTV(25),YMORTN(500),YMORTG(500)
      COMMON TNONMER(25),GNONMER(25),VNONMER(25)
      VGROW=0.
      VGROW1=0.
C-----------------------------------------------------------------------
C               ASSUME FIRST COMERCIAL THINNING AT FIRST STAGE
C-----------------------------------------------------------------------
      ADJ1=(405.-TBASE)/400.
      GLIMIT=10**(3.3446-0.3328*ALOG10(TREE))
      HT=10**(0.1567-15.673/(TAGE-10.)+ALOG10(SITE))
      TMPAGE=TAGE-9.5
      DO 10 I=1,10
      NN=TMPAGE-30.+2.
      IF(NN.LT.1) NN=1
      IF(NN.GT.200) NN=200
      BHAGE=TMPAGE-13.22+0.033*SITE
      DHT=10**(1.7141+ALOG10(SITE)-15.673/TMPAGE-2.*ALOG10(TMPAGE))
      HT=HT+DHT
      VG=10**(-0.0282+0.7917*ALOG10(HT))
C-----------------------------------------------------------------------
C               CALCULATES GROSS GROWTH
C-----------------------------------------------------------------------
      DVA=1.12+0.0105*TMPAGE-0.00005*TMPAGE**2
      IF(TMPAGE.GT.105.) DVA=10**0.22304
      DVOL=10**(ALOG10(2.3026)+ALOG10(12.4083/TMPAGE**2+.4352/BHAGE**
     $1.25)+ALOG10(DVA)+1.9628-12.4083/TMPAGE-1.7408/BHAGE**0.25+
     $1.3176*ALOG10(SITE))
      DVOL=DVOL*ADJ1
      TMPVOL=VGROW+DVOL+V
      G=TMPVOL/VG
      GMERCH=G-YMORTG(NN)
      CR=GMERCH/GLIMIT
      ADJ2=1.-16.*(CR-0.5)**4
      DVOL=DVOL*ADJ2
      VGROW=VGROW+DVOL
C-----------------------------------------------------------------------
C               CALCULATES NET GROWTH
C-----------------------------------------------------------------------
      DVOL1=10**(ALOG10(2.3026)+ALOG10(12.4083/TMPAGE**2+.4352/BHAGE**
     $1.25)+1.9628-12.4083/TMPAGE-1.7408/BHAGE**.25+1.3176*ALOG10(SITE))
      DVOL1=DVOL1*ADJ1
      TMPVOL1=VGROW1+DVOL1+V
      G1=TMPVOL1/VG
      GMERCH1=G1-YMORTG(NN)
      CR1=GMERCH1/GLIMIT
      ADJ21=1.-16.*(CR1-0.5)**4
      DVOL1=DVOL1*ADJ21
      VGROW1=VGROW1+DVOL1
      TMPAGE=TMPAGE+1.
10    CONTINUE
C-----------------------------------------------------------------------
C               CALCULATE MERCHANTABLE MORTALITY
C               AND MERCHANTABLE LIVE TREES.
C-----------------------------------------------------------------------
      N=(TBASE-30.)/10.+M
      IF(N.LT.1) N=1
      IF(N.GT.24) N=24
      VMORT1=VGROW-VGROW1
      VMER=V+VGROW1-VNONMER(N+1)
      IF(TESTP.EQ.1.) VMER=VMER+VMORT1
      GMORT1=(V+VGROW)/VG-(V+VGROW1)/VG
      GMER=(V+VGROW)/VG-GMORT1-GNONMER(N+1)
      IF(TESTP.EQ.0.) RETURN
      VMER=V+VGROW-VNONMER(N+1)
      GMER=VMER/VG
      GMORT1=0.
      VMORT1=0.0
      RETURN
      END
C
C*****************************************************************
C              FUNCTION REVNOW                                   *
C*****************************************************************
      FUNCTION REVNOW(VOL,D,S,R1,TAGE,HAUL)
      IF(VOL.LT.0)VOL=.1
C-----------------------------------------------------------------------
C              CALCULATES STUMP TO TRUCK LOGGING COST
C-----------------------------------------------------------------------
      IF(D.GT.6.05) GOTO 5
      COST=7790.9*VOL**(-0.2834)
      GOTO 150
    5 IF(D.GT.7.63) GOTO 10
      IF(VOL.GT.1000.) GOTO 7
      C1=7790.9*VOL**(-.2834)
      C2=4954.7*VOL**(-.2726)
      DELTA=(C1-C2)/(7.63-6.05)
      COST=C1-DELTA*(D-6.05)
      GOTO 150
    7 C1=7800.8*VOL**(-.2539)
      C2=7187.5*VOL**(-.2891)
      DELTA=(C1-C2)/(7.63-6.05)
      COST=C1-DELTA*(D-6.05)
      GOTO 150
   10 IF (D.GT.9.23) GO TO 20
      IF(VOL.GT.1000.) GOTO 15
      C1=4954.7*VOL**(-.2726)
      C2=3768.0*VOL**(-.2662)
      DELTA=(C1-C2)/(9.23-7.63)
      COST=C1-DELTA*(D-7.63)
      GOTO 150
   15 IF(VOL.GT.2000.) GOTO 18
      C1=7187.5*VOL**(-.2891)
      C2=6254.8*VOL**(-.3013)
      DELTA=(C1-C2)/(9.23-7.63)
      COST=C1-DELTA*(D-7.63)
      GOTO 150
   18 C1=14353.0*VOL**(-.3782)
      C2=10336.6*VOL**(-.3627)
      DELTA=(C1-C2)/(9.23-7.63)
      COST=C1-DELTA*(D-7.63)
      GOTO 150
```

```
20 IF(D.GT.10.87) GOTO 30
   IF(VOL.GT.1000.) GOTO 25
   C1=3768.0*VOL**(-.2662)
   C2=4375.0*VOL**(-.2833)
   DELTA=(C1-C2)/(10.87-9.23)
   COST=C1-DELTA*(D-9.23)
   GOTO 150
25 IF(VOL.GT.2000.) GOTO 28
   C1=6254.8*VOL**(-.3013)
   C2=4375.0*VOL**(-.2833)
   DELTA=(C1-C2)/(10.87-9.23)
   COST=C1-DELTA*(D-9.23)
   GOTO 150
28 C1=10336.6*VOL**(-.3627)
   C2= 8479.6*VOL**(-.3626)
   DELTA=(C1-C2)/(10.87-9.23)
   COST=C1-DELTA*(D-9.23)
   GOTO 150
30 IF(D.GT.12.31) GOTO 40
   C1=8479.6*VOL**(-.3626)
   C2=6839.5*VOL**(-.3492)
   DELTA=(C1-C2)/(12.31-10.87)
   COST=C1-DELTA*(D-10.87)
   GOTO 150
40 IF(D.GT.13.66) GOTO 50
   C1=6839.5*VOL**(-.3492)
   C2=6276.4*VOL**(-.3498)
   DELTA=(C1-C2)/(13.66-12.31)
   COST=C1-DELTA*(D-12.31)
   GOTO 150
50 IF(D.GT.15.03) GOTO 60
   C1=6276.4*VOL**(-.3498)
   C2=5848.5*VOL**(-.3548)
   DELTA=(C1-C2)/(15.03-13.66)
   COST=C1-DELTA*(D-13.66)
   GOTO 150
60 IF(D.GT.16.19) GOTO 70
   C1=5848.5*VOL**(-.3548)
   C2=4980.1*VOL**(-.3475)
   DELTA=(C1-C2)/(16.19-15.03)
   COST=C1-DELTA*(D-15.03)
   GOTO 150
70 IF(D.GT.17.26) GOTO 80
   C1=4980.1*VOL**(-.3475)
   C2=3765.6*VOL**(-.3238)
   DELTA=(C1-C2)/(17.26-16.19)
   COST=C1-DELTA*(D-16.19)
   GOTO 150
80 IF(D.GT.18.31) GOTO 90
   C1=3765.6*VOL**(-.3238)
   C2=4215.0*VOL**(-.3402)
   DELTA=(C1-C2)/(18.31-17.26)
   COST=C1-DELTA*(D-17.26)
   GOTO 150
90 IF(D.GT.20.25) GOTO 100
```

```
      C1=4215.0*VOL**(-.3402)
      C2=3377.5*VOL**(-.3207)
      DELTA=(C1-C2)/(20.25-18.31)
      COST=C1-DELTA*(D-18.31)
      GOTO 150
  100 IF(D.GT.21.90) GOTO 110
      C1=3377.5*VOL**(-.3207)
      C2=4209.5*VOL**(-.3488)
      DELTA=(C1-C2)/(21.9-20.25)
      COST=C1-DELTA*(D-20.25)
      GOTO 150
  110 COST=4209.5*VOL**(-.3488)
C-----------------------------------------------------------------------
C             CALCULATE CURRENT REVENUE AS POND VALUE LESS
C             LOGGING AND HAUL COST ($/CF)
C-----------------------------------------------------------------------
  150 PONDVAL=(9.91+70.81*D)*(1+R1)**TAGE
      REVNOW=VOL*(PONDVAL-COST/S-HAUL)*0.001
      RETURN
      END
C
C***********************************************************************
C             FUNCTION TARIF                                           *
C***********************************************************************
      FUNCTION TARIF(DIAM)
      TARIF=(0.00497819*DIAM**2)/(0.005454154*(DIAM**2+16.)*(1.0378+
     $1.4967*(0.0134**(DIAM/10.)))-0.174532)
      RETURN
      END
```