Decomposition and Partitioning Methods for Multistage Stochastic Linear Programs

JOHN R. BIRGE
The University of Michigan, Ann Arbor, Michigan
(Received May 1982; accepted September 1984)

Multistage stochastic linear programs model problems in financial planning, dynamic traffic assignment, economic policy analysis, and many other applications. Equivalent representations of such problems as deterministic linear programs are, however, excessively large. This paper develops decomposition and partitioning methods for solving these problems and reports on computational results on a set of practical test problems.

MANY practical problems require that decisions be made periodically over time. These problems can often be formulated as multistage linear programs. The resulting programs sometimes have a staircase structure that can be exploited by decomposition (Glassey 1971, 1973; Ho and Manne 1974), basis factorization (Fourer 1979), partitioning (Rosen 1963), and other computational techniques.

Uncertainty about some parameters of the program can, however, cause difficulties. The random coefficients are often replaced by their expected values, but the solution of the corresponding expected value program might not be optimal for the stochastic program. The error resulting from replacing random variables by mean values can indeed be quite large (Kallberg, White and Ziemba 1982). Using a single deterministic value other than the mean can also lead to large inaccuracies (Birge 1982). In these cases, a stochastic program allowing for several possible coefficient values must be solved to obtain an optimal solution. The deterministic equivalent linear program for this stochastic program is usually very large, and standard solution procedures are costly.

This paper gives two methods for solving these large scale programs by exploiting their structure. It first presents the methods and then compares them with the standard simplex method on a set of test problems.

The first method is an outer linearization decomposition approach

Subject classification: 635 large scale systems programming, 655 stochastic programming.

Operations Research
Vol. 33, No. 5, September–October 1985
0030-364X/85/3305-0989 $01.25

that extends the two stage L-shaped method of Van Slyke and Wets (1969). The procedure approximates the convex objective term in the stochastic program by successively appending supporting hyperplanes. In multistage programs, the supports are found by optimizing a sequence of nested problems of the same form. Each problem has a linear objective term for current costs and a convex objective term representing future costs. The solution procedure is described in detail in Section 2.

As shown in Section 2, the optimization problems solved to obtain the supporting hyperplanes in the decomposition method are degenerate. As a consequence, the method might append many similar supports, a difficulty that motivated the development of the second method, a piecewise partitioning strategy. This approach determines the optimal first period decision by partitioning the feasible region of the equivalent convex program into regions of linearity for the objective. A linear optimization is performed on adjacent regions until no adjacent region yields an improved solution. The regions and structure of the objective are again found by optimizing similar nested problems that determine optimal future decisions given current decisions and outcomes. The partitioning strategy is described in Section 3. Section 4 applies both algorithms on a set of test problems.

1. PROBLEM FORMULATION

The stochastic linear program with fixed recourse was first formulated by Beale (1955) and Dantzig (1955). A multistage version is

$$\begin{aligned}
&\text{minimize} \\
&\quad z = c_1x_1 + E_{\xi_2}[\min\ c_2x_2 + \cdots + E_{\xi_t}[\min\ c_Tx_T]\cdots] \\
&\text{subject to} \\
&\quad A_1x_1 = b_1, \\
&\quad -B_1x_1 + A_2x_2 = \xi_2, \\
&\qquad\qquad \ddots \qquad\qquad \vdots \\
&\qquad\qquad -B_{T-1}x_{T-1} + A_Tx_T = \xi_T, \\
&\quad x_t \geq 0, \quad t = 1, \cdots, T, \quad \xi_t \in \Xi_t, \quad t = 2, \cdots, T,
\end{aligned} \tag{1}$$

where c_t is a known vector $\mathbf{R}^{n_t}$ for $t = 1, \cdots, T$, b_1 is a known vector in $\mathbf{R}^{m_1}$, ξ_t is a random m_2-vector defined on the probability space $(\Xi_t, \mathbf{F}_t, F_t)$ for $t = 2, \cdots, T$, and A_t and B_t are correspondingly dimensioned known real-valued matrices. Transposes have been eliminated for simplicity. "E_{ξ_t}" represents the mathematical expectation with respect to ξ_t.

The vector x_1 in (1) represents decisions made in the first period. These decisions are nonanticipative, or independent, of any specific realization of the random outcomes. Future decision vectors x_t depend on previous outcomes, $\xi_2, \cdots, \xi_t$, and decisions, $x_1, \cdots, x_{t-1}$. The problem is to determine a first period decision that minimizes the sum of current costs, c_1x_1, and the expected values of all future costs until some time T.

Several solution methods have been proposed for program (1) with two periods and finite distributions. Dantzig and Madansky (1961) showed how the dual of this problem could be solved using Dantzig-Wolfe (1960) decomposition. Kall (1979) and Strazicky (1980) presented another dual method based on basis factorization. Van Slyke and Wets, however, gave a method for solving the primal problem using outer linearization, or Benders' decomposition (Benders 1962). Section 2 presents an extension of this algorithm for multiple periods.

Other methods have been applied to the simple recourse problem in which $A_T = [I \vdots -I]$. For two periods, Wets (1974, 1975, 1983) has presented methods based on discrete distributions and splines. For three periods, Everitt and Ziemba (1979) showed how a variant of the Frank-Wolfe algorithm could solve the problem. Numerical results from a modification of this algorithm appear in Kallberg and Ziemba (1981). Applications include many problems in dynamic economic planning (Fox, Sengupta and Thorbecke 1966; Theil 1964) and financial planning (Kallberg, White and Ziemba; Kusy 1978; Bradley and Crane 1972, 1976).

For multistage problems, an appropriate horizon length, T, is difficult to determine. The horizon should be distant enough for present decisions to reflect the future accurately, but short enough to allow for efficient computation. Grinold (1980) has shown that, under appropriate conditions, finite horizon solutions can be found that are close to an infinite horizon optimum. He has also presented methods based on steady state future decisions and resource values that can diminish the end effects associated with finite horizons (Grinold 1979, 1983). In other papers, Grinold has formulated the multistage stochastic program as a finite Markov chain (Grinold 1976) and as equivalent primal and dual optimization problems (Grinold 1979).

Beale, Forrest and Taylor (1980) consider a multistage production model. They show that a backward recursion with an appropriate approximation can obtain results that are very similar to exact dynamic programming solutions. They use a piecewise linear approximation that might crudely represent the actual distribution, but might still produce a "fairly" accurate solution.

The general multistage problem (1) can be modeled as a dynamic program with stages $1, \cdots, T$ and states $y_t = B_tx_t$ for $t = 1, \cdots, T-1$. An optimal solution to (1) can in principle be found by backward

recursion using

$$z_t(y_{t-1}) = E_{\xi_t}[\zeta(y_{t-1}, \xi_t)], \tag{2}$$

where

$$\begin{aligned} \zeta(y_{t-1}, \xi_t) = \min_{(x_t, y_t)} & c_t x_t + z_{t+1}(y_t) \\ \text{subject to} \quad & A_t x_t = \xi_t + y_{t-1}, \\ & B_t x_t = y_t, \\ & x_t \geq 0. \end{aligned} \tag{3}$$

When $t = T$, $B_T = 0$ and $z_{T+1}(0) = 0$.

An optimal set of first period decisions, x_1^*, is found by solving (3) with b_1 replacing ξ_1 for $t = 1$. The dimension of y_t might, however, make direct implementation of this procedure impossible. The methods given in the following sections use outer linearization and piecewise linear partitioning to approximate $z_{t+1}(y_t)$. This simplification makes the recursion in (3) calculable.

2. A NESTED BENDERS' DECOMPOSITION METHOD

Many decomposition methods can be viewed as employing the fundamental strategies of *outer linearization* (e.g., Benders) or *inner linearization* (e.g., Dantzig and Wolfe). Geoffrion (1970) classified large-scale programming algorithms according to these problem manipulations, and Kallio and Porteus (1977) discussed their application to dynamic linear programs. The Ho and Manne application of inner linearization to successive stages of dynamic linear programs is *nested decomposition* as in Glassey (1971, 1973).

Ho and Manne's procedure requires a time consuming Phase III to obtain primal variable values. It is also particularly difficult to adapt to stochastic programs because it begins with the last-period problem as the first master problem. In stochastic programs, the large number of these last-period problems would render Ho and Manne's approach impractical. Outer linearization procedures, such as the L-shaped method (Van Slyke and Wets), are preferred because they can solve a group of last-period problems together.

The L-shaped method can be extended to the multi-stage method presented in this section. The approach maintains primal variable values and avoids a Phase III procedure. It can be alternatively described as applying inner linearization to the dual of (3). It is based on the following result:

PROPOSITION 1. *The mathematical program* (3) *is equivalent to*

$$\text{minimize} \quad c_t x_t + Q_{t+1}(x_t) \tag{4.1}$$

$$\text{subject to} \quad A_t x_t = \xi_t + B_{t-1} x_{t-1}, \tag{4.2}$$

$$D_t^l x_t \geq d_t^l, \quad l = 1, \cdots, q, \tag{4.3}$$

$$x_t \geq 0, \tag{4.4}$$

for some convex function $Q_{t+1}(x_t)$, n_t*-vectors* D_t^l, *constants* d_t^l, *and* $q \leq m_{t+1}$.

Proof. See Wets (1966), Proposition 6.

Feasibility cuts (4.3) are used to induce a feasible solution at stage $t + 1$. They can be constructed sequentially as in the L-shaped method of Van Slyke and Wets. Outer linearization is then applied to an alternative formulation of (4):

$$\text{minimize} \quad c_t x_t + \theta_t \tag{5.1}$$

$$\text{subject to} \quad A_t x_t = \xi_t + B_{t-1} x_{t-1}, \tag{5.2}$$

$$D_t^l x_t \geq d_t^l, \quad l = 1, \cdots, q, \tag{5.3}$$

$$-Q_{t+1}(x_t) + \theta_t \geq 0, \tag{5.4}$$

$$x_t \geq 0. \tag{5.5}$$

PROPOSITION 2. *Constraint* (5.4) *in problem* (5) *for* $t = 1, \cdots, T$ *can be replaced by linear constraints*

$$E_t^l x_t + \theta_t \geq e_t^l, \quad l = 1, \cdots, p, \tag{6}$$

where $p \leq m_{t+1} + 1 < +\infty$, E_t^l *is an* n_t*-vector and* e_t^l *is a constant.*

Proof. See Van Slyke and Wets for a proof of this result. The bound on p is due to Murty (1968) and is a straightforward result of the basic property of optimal solutions to convex minimization problems with linear constraints.

Program 5 is solved by forming a relaxed problem excluding (5.3) and (5.4) and by successively adding constraints (5.3) and (6) until a solution, (x_t^*, θ_t^*), of the relaxed problem is found that satisfies $Q_{t+1}(x_t^*) \leq \theta_t^*$. For the algorithm to be used, we must assume that a finite number, K_t, of *scenarios* is possible for each t. The scenarios can represent discretizations of continuous random variables as in Dantzig and Madansky and Van Slyke and Wets. Program 5 must be solved for each scenario. The scenarios consist of all possible realizations of the vectors from periods one through t. For example, in a three period problem with two realizations of the random vector in each of periods two and three, there are

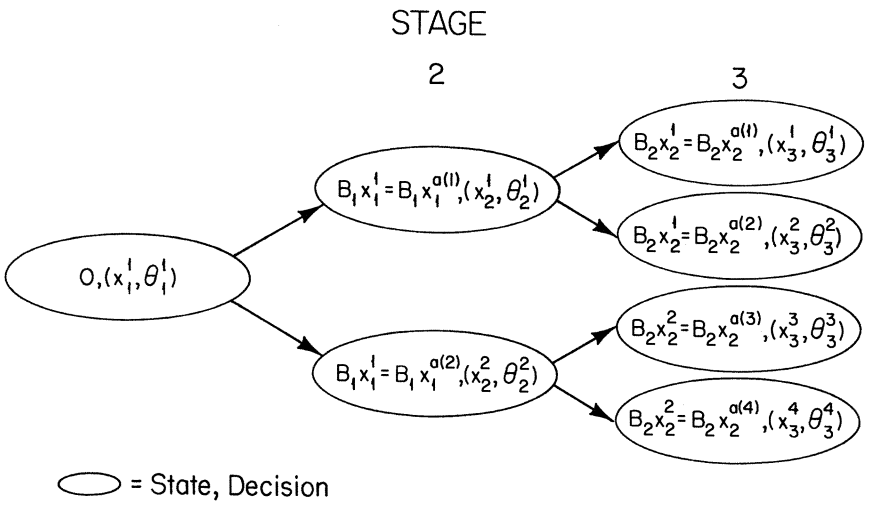

Figure 1. Stages and decisions for a three-period problem.

four scenarios in period three. Each period-three scenario has an *ancestor* scenario in period two, and each period-two scenario has two *descendant* scenarios in period three (see Figure 1). In general, the random vector for scenario j in period t is ξ_t^j for $j = 1, \cdots, K_t$. Many of these values might, however, be identical. The relaxation of (5) for period t and scenario j after r_t^j constraints (5.3) and s_t^j constraints (6) have been added is

$$\text{minimize} \quad c_t x_t^j + \theta_t^j \tag{7.1}$$

$$\text{subject to} \quad A_t x_t^j = \xi_t^j + B_{t-1} x_{t-1}^{a(j)} \tag{7.2}$$

$$D_t^{l,j} x_t^j \geq d_t^{l,j}, \quad l = 1, \cdots, r_t^j \tag{7.3}$$

$$E_t^{l,j} x_t^j + \theta_t^j \geq e_t^{l,j}, \quad l = 1, \cdots, s_t^j \tag{7.4}$$

$$x_t^j \geq 0, \tag{7.5}$$

where $a(j)$ is the ancestor scenario of j in period $t-1$, and $x_{t-1}^{a(j)}$ is the current solution of the $a(j)$ scenario problem in period $t-1$. The multistage nested decomposition first obtains a feasible solution for (7) for all t and j and then solves (5) sequentially from periods T to one using the relaxation (7).

Nested Decomposition for Stochastic Programming Algorithm (NDSPA)

Step 0. Solve (7) for $t = 1$ where $\theta_1 = 0$, $r_1 = s_1 = 0$, and (7.2) is replaced by

$$A_1 x_1 = b_1. \tag{8}$$

(The scenario index j has been dropped for the period 1 problem.) Set $\theta_t^j = 0$ and $r_t^j = s_t^j = 0$ in problem (7) for all t and scenarios j at t. (The r_t^j and s_t^j indices are updated whenever a constraint (7.3) or (7.4) is added to (7).)

Step 1. If the period 1 problem (7) is infeasible, STOP. The problem is infeasible.

Otherwise, let $\bar{x}_1$ be the current optimal solution of (7) for $t = 1$. Use $\bar{x}_1$ as an input in (7.2) for $t = 2$. Solve (7) for $t = 2$ and all $\xi_2^j, j = 1, \cdots, K_2$.

If any period two problem is infeasible, then add a feasibility constraint (7.3) to (7) for $t = 1$, re-solve for $t = 1$, and return to 1.

Otherwise, let $t = 2$ and go to 2.

Step 2. a) Let the current period t optimal solutions be $\bar{x}_t^j$ for $j = 1, \cdots, K_t$. Solve (7) for $t + 1$ and all $j = 1, \cdots, K_{t+1}$ using the appropriate ancestor solution $\bar{x}_t^j$ in (7.2).

b) If any period $t + 1$ problem is infeasible, add a feasibility constraint to the corresponding ancestor period t problem and re-solve that problem.

If the period t problem is infeasible, let $t = t - 1$.

If $t = 1$, go to 1.

Otherwise, return to 2a.

Otherwise, return to 2a.

Otherwise, all period $t + 1$ problems are feasible.

If $t \leq T - 2$, let $t = t + 1$ and return to 2a.

Otherwise ($t = T - 1$), remove any remaining $\theta_\tau^j = 0$ restrictions for all periods τ and scenarios j at τ and, for each of these, let the current value of θ_τ^j be $\overline{\theta}_\tau^j = -\infty$. Go to 3.

Step 3. a) Find $E_t^{l,j}$ and $e_t^{l,j}$ for a new constraint (7.4) at each scenario t problem (7) using the current period $t + 1$ solutions. The vector $E_t^{l,j}$ is calculated as $-\sum_{d(j)} \pi_{t+1}^{d(j)} B_t$ and $e_t^{l,j} = \sum_{d(j)} \pi_{t+1}^{d(j)} \xi_{t+1}^{d(j)}$ where the $d(j)$ scenarios are the descendants of j and $\pi_{t+1}^{d(j)}$ is an optimal dual vector in the $d(j)$ descendant problem.

b) If some j satisfies

$$\overline{\theta}_t^j < e_t^{l,j} - E_t^{l,j} \bar{x}_t^j, \tag{9}$$

then add the new constraint (7.4) to each period t problem (7) for which (9) holds. Solve each period t problem (7). Use the resulting solutions $(\bar{x}_t^j, \overline{\theta}_t^j)$ to form (7.2) for the corresponding descendant period $t + 1$ problems (7) and re-solve each period $t + 1$ problem (7).

If $t < T - 1$, let $t = t + 1$ and go to 2a.

Otherwise, return to 3a.

Otherwise, $\bar{\theta}_t^j = e_t^{Lj} - E_t^{Lj}\,\bar{x}_t^j$ for all scenarios j at t.
If $t > 1$, let $t = t - 1$ and return to 3a.
Otherwise, STOP. The current solutions $\bar{x}_\tau^j$, $\tau = 1, \cdots, T$ form an optimal solution of (1).

Steps 1 and 2 of NDSPA form a *forward pass* through all periods to rapidly locate a feasible solution of (1). Step 3 is a *backward pass* to solve the dynamic programming recursion defined by (2) and (3). This optimization phase might take much longer than the forward pass since many marginal changes in the last periods could be necessary before the initial period solutions are even considered. In general, the forward pass proceeds from Step 1 to Step 2 and iterates in Step 2 until a feasibility cut is sent back to period 1 or a feasible solution is found for period T. The backward pass starts from Step 3 and proceeds to Step 2 when an optimality cut is placed in any period before $T - 1$. The general path in terms of NDSPA steps might require the following steps:

Forward pass: $1 \Rightarrow 2 \Rightarrow 2 \Rightarrow 1 \Rightarrow 2 \cdots \Rightarrow 2 \Rightarrow 3 \Rightarrow$

Backward pass: $3 \Rightarrow 2 \Rightarrow 2 \Rightarrow 3 \Rightarrow 2 \Rightarrow 2 \Rightarrow$

$\cdots \Rightarrow 2 \Rightarrow 3 \Rightarrow 3 \Rightarrow \cdots \Rightarrow 3 \Rightarrow$ optimal.

Remarks

1. Unboundedness can be resolved by the procedure in Van Slyke and Wets. Their method considers the half-line along which the objective in (7) goes to $-\infty$ as the sum of a basic solution of (7) and multiples of a homogeneous solution of (7). The descendant problems are solved with the basic and homogeneous solutions as inputs in the right-hand side of (7.2) to determine whether a new feasibility cut can be added to the ancestor problem. In practice, unboundedness is avoided by giving each variable a finite upper bound.
2. The algorithm proceeds back from t to $t - 1$ after a finite number of steps since there are finitely many scenarios in each period. The forward pass to obtain feasibility also proceeds forward in a finite number of steps. Hence, the algorithm converges finitely.
3. The constraint matrix A_t is the same for every scenario $j = 1, \cdots, K_t$. A single basis factorization could, therefore, be stored for many scenarios. *Bunching* (see Wets 1983) or *sifting* (Garstka and Rutenberg 1973) procedures are used to pass efficiently through the set of random vectors to find all ξ_t^j for which a basis is optimal. Our implementation of NDSPA uses bunching for period T. Additional constraints in earlier periods make a repeated basis difficult to obtain, although common $m_t \times m_t$ working sub-bases could be found.
4. Degeneracy might lead to many repeated solutions of problems (7) for a single period t scenario. Abrahamson (1980) observed this behavior

in deterministic multistage linear programs and presented a method for avoiding degeneracy. His procedure does not directly apply to stochastic programs, but an alternative is given in Birge (1980) that creates an intermediate problem between an ancestor problem (7) and its descendant problems. By design, the proposed partitioning method of Section 3 also decreases the number of optimizations of problems (7).

The following propositions describe the degeneracy discussed in Remark 4. They are proven in Birge (1980) and follow directly from the basis property of an optimal solution of (1).

PROPOSITION 3. *Let scenario j' at period $t + 1$ generate a feasibility constraint l' of type* (7.3) *for the ancestor scenario j at period t. If constraint l' of type* (7.3) *is binding for some solution, $\bar{x}_t^j$, of* (7) *for scenario j at period t, then every basic feasible solution of problem* (7) *for scenario j' at period $t + 1$ with $\bar{x}_t^j$ input in* (7.2) *is degenerate.*

PROPOSITION 4. *Let the set of descendant scenarios of j at period t be J'. If two constraints of type* (7.4) *are binding at some solution $(\bar{x}_t^j, \bar{\theta}_t^j)$ of* (7) *for scenario j at period t, then every set of optimal solutions of problems* (7) *for period $t + 1$ and descendants j' that has $\bar{x}_t^j$ input in* (7.2) *and that produces $E_t^{l,j}$ and $e_t^{l,j}$ that do not satisfy* (9) *includes a degenerate solution for some $j' \in J'$.*

3. A PIECEWISE LINEAR PARTITIONING METHOD

The independence of the scenario j problem (7) at period t from its descendant problems j' at period $t + 1$ leads to the degeneracy problem of NDSPA. The same set of basic variables might occur in several consecutive optimizations of the period t problem. The partitioning method of this section attempts to avoid this situation by forcing a basis change.

NDSPA also does not explicitly maintain primal feasibility throughout the backward pass. As a result, it might be difficult to bound the optimal solution value before the algorithm has terminated, although feasibility always holds in proceeding from Step 2 to Step 3 at which time the objective increases monotonically. The partitioning method avoids infeasibility after the algorithm has found an initial primal feasible solution.

The finite scenario assumptions and notation from the NDSPA discussion apply to the partitioning method. The basic idea of the method is similar to Rosen's algorithm. The method minimizes a convex function at some point in a subregion of the feasible region. It then searches all subregions adjacent to the current point for a lower value. If none is found, then the procedure stops, and the current point is optimal.

Subregions in our method correspond to regions of linearity of $Q_{t+1}(x_t)$ in (4). Consider scenario j at period $T-1$. The scenario index j will be dropped from the decision vector x^j_{T-1} to simplify notation. Let the descendants of j be scenarios $j' = 1, \cdots, k$, and let a corresponding set of dual feasible bases in problem (3) for period T be $H^1, \cdots, H^k$. If x_{T-1} satisfies

$$(H^i)^{-1}(\xi^i_T + B_{T-1}x_{T-1}) \geq 0 \tag{10}$$

for $i = 1, \cdots, k$, then

$$Q_T(x_{T-1}) = \tilde{q}_T x_{T-1} + \tilde{C}_{T-1}, \tag{11}$$

where the n_{T-1}-vector $\tilde{q}_T$ and the constant $\tilde{C}_{T-1}$ depend on $H^1, \cdots, H^k$ and $\xi^1_T, \cdots, \xi^k_T$. These parameters are formed directly by multiplying the left-hand side of (10) by the corresponding objective coefficients and by summing each of these terms weighted by the probability of each scenario. Hence, the restriction of problem (4) to linear subregions yields the program

$$\text{minimize} \quad z_{T-1} = \tilde{c}_{T-1}x_{T-1} + \tilde{C}_{T-1} \tag{12.1}$$

$$\text{subject to} \quad A_{T-1}x_{T-1} = \xi_{T-1} + B_{T-2}\bar{x}_{T-2}, \tag{12.2}$$

$$G^i_{T-1}x_{T-1} \geq g^i_{T-1}, \quad i = 1, \cdots, k, \tag{12.3}$$

$$x_{T-1} \geq 0, \tag{12.4}$$

where

$$\tilde{c}_{T-1} = c_{T-1} + \tilde{q}_T,$$

$$G^i_{T-1} = (H^i)^{-1}B_{T-1},$$

$$g^i_{T-1} = -(H^i)^{-1}\xi^i_T,$$

and $\bar{x}_{T-2}$ is the current ancestor period $T-2$ solution. Let the optimal solution of (12) be x^0_{T-1} with value z^0_{T-1}. The solution x^0_{T-1} solves (4) for $T-1$ if none of the constraints (12.3) is binding. A binding constraint (12.3) for some i corresponds to a degeneracy in the corresponding period T problem. An adjacent optimal basis can be found in descendant problem i, and (12) can be reformulated on an adjacent region of linearity. The objective value of (12) on the adjacent region is $\bar{z}^0_{T-1}$ and the corresponding solution is $\bar{x}^0_{T-1}$.

The partitioning method lists all binding constraints (12.3) for x^0_{T-1} and proceeds through the list by solving an adjacent problem (12) until some $\bar{z}^0_{T-1}$ is found satisfying $\bar{z}^0_{T-1} < z^0_{T-1}$. If none is found, x^0_{T-1} is again optimal for (4). When $\bar{z}^0_{T-1} < z^0_{T-1}$, then $z^1_{T-1} = \bar{z}^0_{T-1}$ and $x^1_{T-1} = \bar{x}^0_{T-1}$. The process is repeated for binding constraints of x^1_{T-1}. The method

continues to increment the index of z_{T-1} and x_{T-1} whenever a better solution is found. When z^M_{T-1} is found so that no adjacent $\bar{z}^M_{T-1} < z^M_{T-1}$, then x^M_{T-1} is optimal for (4). The method converges finitely because $z^0_{T-1} > \cdots > z^M_{T-1}$ and only a finite number of basis sets are possible in the period T problems.

Substantial storage reductions are possible if only distinct bases are used in constraints (12.3). G^i_{T-1} might be the same for several descendant scenarios i. These constraints can be combined by defining g^i_{T-1} to be the maximum in each component among the descendants associated with a single basis.

The full multistage partitioning algorithm follows the same fundamental flow as NDSPA. First, a forward pass obtains a feasible solution and, then, a backward pass achieves optimality. Linear subregions are found for programs of type (12) for periods $t < T-1$ by constructing constraints (12.3) to ensure primal feasibility in all future periods.

Piecewise Linear Partitioning Algorithm (PLPA)

Step 1. Follow Steps 0, 1 and 2 of NDSPA to obtain a feasible solution of (1). Let $t = T$, $j = 0$, and $M = -1$.

Step 2. If $t = 0$, STOP. The current solution $\hat{x}_1$ is an optimal first period decision in (1).

Otherwise, let $M = M + 1$.

If $M = 0$, let $j = j + 1$ and solve (12) for scenario j and period t to obtain z^0_t and x^0_t. Re-solve (12) on adjacent regions of linearity until $\bar{z}^0_{T-1}$ is found satisfying $\bar{z}^0_{T-1} < z^0_t$. Let $z^1_t = \bar{z}^0_{T-1}$ and $x^1_t = \bar{x}^0_{T-1}$. Return to 2.

Otherwise, if $z^M_t = z^{M-1}_t$, let $M = -1$.

If $j = K_t$, let $t = t - 1$, $j = 1$, and $M = -1$. Return to 2.

Otherwise, let $j = j + 1$ and return to 2.

Otherwise, re-solve (12) for j and t on adjacent regions of linearity until $\bar{z}^M_t$ is found satisfying $\bar{z}^M_t < z^M_t$. Let $z^{M+1}_t = \bar{z}^M_t$, $x^{M+1}_t = \bar{x}^M_t$ and return to 2.

Remarks

1. Unboundedness in any problem (12) implies (1) is unbounded. In practice, bounded variables are assumed.
2. The algorithm converges finitely because each period t problem (4) is solved finitely by the partitioning method.
3. Primal feasibility is maintained throughout Step 2 of PLPA. The restrictions in (12.3), however, steadily increase as t decreases.

PLPA's advantages over NDSPA are increased connections between ancestor and descendant problems and the preservation of primal feasi-

bility. The problems (12) at each period t are, however, larger than the problems (7) solved by NDSPA. A comparison of PLPA and NDSPA on a set of test problems follows.

4. COMPUTATIONAL RESULTS

Two FORTRAN programs, NDST3 and PCST2 (Birge 1981), were written to implement NDSPA and PLPA, respectively. Linear programs of types (7) and (12) were solved using LPM-1 (Pfefferkorn and Tomlin 1976).

NDST3 currently solves problems with up to four periods. For a four-period problem, the program allows up to three realizations of the right-hand side in the second period followed by up to three descendants for each of these scenarios in the third period. In the last period, the program allows up to 125 different realizations of the right-hand side vector (e.g., three random variables with five realizations each) for each of the nine period-three ancestor scenarios. This choice results in up to 1125 period-four scenarios.

The size of each problem (7) solved by NDST3 does not exceed 350 rows, 600 columns, or 3000 nonzero elements. The number of last period realizations of the right-hand side vector could be increased without significantly enlarging storage requirements because NDST3 stores distinct bases only for the last period. For periods $t < T$, each problem (7) is solved separately. Any increase in the number of these intermediate period problems would, therefore, proportionally increase storage. Out of core storage might, however, allow for more possibilities.

PCST2 currently applies only to two stages. Attempts to implement the code for more periods showed that multistage storage needs would be much greater than the two-stage storage requirements. PCST2 allows up to 125 realizations of the right-hand side vector. Each problem (12) solved contains at most 350 rows, 600 columns and 3000 nonzero elements. PCST2 uses bunching to solve second period problems efficiently and could, therefore, be easily expanded to allow for more right-hand side realizations.

PCST2 and NDST3 can be compared with existing codes for simple recourse problems. Kallberg and Kusy (1976) have coded Wets's (1974, 1983) algorithm for two period ($T = 2$) simple recourse problems. Their code allows up to 70 random variables with eight realizations each in a program with up to 220 rows. This problem size could again be easily expanded to allow for more possible right-hand sides.

NDST3 and PCST2 were tested on stochastic versions of problems from Ho and Loute (1981), which are practical deterministic multi-stage problems with staircase structure, as in the constraint matrix of (1). For comparison, each problem had three periods and eight right-hand side realizations. The full deterministic equivalent problem to (1) could then

be easily solved to provide a check on the solution values. Note, however, that NDSPA and PLPA would probably be most useful in problems for which a full deterministic equivalent linear program cannot be solved directly.

The test problems have the following applications.

Problem	*Application*
SC205	Economic development
SCAGR7	Agricultural planning
SCRS8	Dynamic energy modeling
SCSD8	Structural design optimization
SCTAP1	Dynamic traffic assignment
SCFXM1	Production scheduling.

Size characteristics of these problems appear in Table I. The dimensions of the deterministic equivalent program appear in the columns headed "Totals." Descriptions and problem origins appear in Ho and Loute.

The optimal values in Table I were obtained by solving the full deterministic equivalent program using the mathematical programming code, MINOS (Murtagh and Saunders 1978), on The University of Michigan's Amdahl 470/V8 computer. Mean values were also substituted for the random variables to obtain the *mean value problem* also solved by MINOS. The CPU times in seconds (excluding input and output) and iterations for MINOS to solve each problem ("Mean" and "Full") appear in Table II. Table II shows the significant increase in times from the mean value problem to the stochastic problem. The ratio of stochastic problem time to mean value problem time on MINOS ranges from 11.6 for SCRS8 to 29.6 for SCTAP1.

The test problems were each solved by NDST3 and PCST2 on the Amdahl 470/V8 using the standard FORTRAN-G compiler without optimization (The University of Michigan Computing Center 1980). Each problem has three periods, but an equivalent two-period problem can be formulated by making the second-period objective include the expected value of third period objective. The second period of the two-period equivalent problem then determines both period-two and period-three decisions in the original problem. The results from NDST3 on both the two-period and three-period equivalent problems appear in Table II. The results of PCST2 on the two-period equivalent problem also appear in Table II. In general, the number of cuts (12.3) on problem (12) for PCST2 becomes quite large as t decreases. For the problems in Table II, the maximum occurs in SCAGR7 in which $4m_2$, or 120, constraints (12.3) are added to the 15 constraints (12.2). CPU seconds in Table II do not include problem input or solution output. Table II also gives the number of simplex pivots and the number of distinct problems (7) for NDST3 and problems (12) for PCST2.

TABLE I

PROBLEM STATISTICS

Problem	Area	Within Period				Totals				Optimal Value
		Rows		Columns		Rows	Columns	Nonzero elements	Density[a]	
		Max	Min	Max	Min					
SC205	Economic Development	13	11	14	11	190	380	701	0.97	−60.4
SCAGR7	Agricultural Planning	19	15	20	20	320	660	1693	0.80	−832903.5
SCRS8	Dynamic Energy Modeling	31	28	38	37	254	342	1133	1.3	123.4
SCTAP1	Dynamic Traffic Assignment	30	30	48	48	511	817	3814	0.91	240.5
SCSD8	Structural Design Optimization	10	10	70	70	171	1191	4867	2.4	16.0
SCFXM1	Production Scheduling	92	82	126	99	1277	1915	7952	0.33	2877.6

[a] Percent of elements that are nonzero.

TABLE II

RESULTS

Problem	MINOS				NDST3 (Two-Period Equivalent)			NDST3 (Three-Period Equivalent)			PCST2		
	Mean		Full										
	CPUs[a]	Simplex pivots	CPUs	Simplex pivots	CPUs	Simplex pivots	Problems (7) solved	CPUs	Simplex pivots	Problems (7) solved	CPUs	Simplex pivots	Problems (12) solved
SC205	0.07	20	1.31	118	0.07	24	8	0.26	28	25	0.13	24	7
SCAGR7	0.27	46	5.37	269	0.49	100	12	0.61	117	49	0.93	69	11
SCRS8	0.22	37	2.56	203	0.11	28	4	0.29	23	19	0.25	11	3
SCSD8	0.39	55	11.23	381	0.15	11	4	0.39	52	19	0.34	11	3
SCTAP1	0.35	49	10.37	337	0.43	64	4	0.76	92	35	0.71	63	4
SCFXM1	5.48	316	138.56	1742	—[b]	—[b]	—[b]	—[c]	—[c]	—[c]	—[b]	—[b]	—[b]

[a] CPU seconds excluding problem input and solution output.

[b] Exceeded nonzero element limit after 40 passes and over 800 iterations. Last solution was 25% from optimal.

[c] Exceeded nonzero element limit after 49 passes and over 1,000 iterations. Last solution was 35% from optimal.

In the first five test problems, NDST3 with the two-period formulation had the lowest CPU times. The ratio of these times to the mean value times ranged from 0.4 for SCSD8 to 1.8 for SCAGR7. This experience is consistent with Kallberg and Kusy's results where these ratios ranged from 1.5 to 2.

The number of distinct problems (7) and (12) solved by NDST3 and PCST2 is another indication of solution difficulty. The lowest numbers of these optimizations occur in problems (SCRS8, SCSD8, SCTAP1) that have optimal solutions with exactly m_1 basic first period columns. NDST3 and PCST2 exploit the decoupling property that this suggests. The optimal solution found for SCAGR7 had $m_1 + 1$ basic first period columns, and the SC205 optimal solution obtained had $m_1 + 2$ basic first period columns. To find these extra columns, the computation times increased.

In the examples, the three-period formulation and partitioning do not appear advantageous. These results on a small set of problems without advanced storage techniques should, however, not be considered definitive. The basic reason for good performance relative to MINOS appears to be simply the elimination of repeated pivot steps for each scenario. Degeneracy did not appear to cause difficulties in the first five problems.

The example SCFXM1 displayed confounding behavior for both NDST3 and PCST2. The forward pass generated many feasibility cuts (7.3). After 23 of these dense cuts were placed on the first period problem, the model exceeded the maximum nonzero element number (3000). Some cuts could have been deleted, but the optimal MINOS solution contained $m_1 + 7$ first period basic columns. This linking indicates that several cuts were necessary. In general, the first period problem could require m_2 completely dense cuts or $m_2 \cdot n_1$ additional nonzero elements.

Excessive nonzero element growth is one of the potential difficulties in decomposition and partitioning procedures. Smaller problems have not demonstrated this property (Birge 1980), but the solution method should incorporate some check on nonzero element growth and should delete slack cuts. Overall, the problems (7) solved by NDST3 are still much smaller than the full deterministic equivalent linear program. For $T = 2$, if $v(1)$, $v(2)$, and $v(3)$ are the numbers of nonzeroes in A_1, B_1, and A_2, respectively, then the deterministic equivalent linear program has $v(1) + K_2(v(2) + v(3))$ nonzeroes in the constraint matrix. The period one problem (7) contains at most $v(1) + n_1\Pi$ nonzero elements, where Π is the number of times (7) is solved. When Π is small as in the first five examples, the number of nonzeroes is also small.

5. SUMMARY

This paper presented two algorithms for multistage stochastic linear programs. The methods are based on decomposition and partitioning

procedures that are common in large scale mathematical programming. Computer codes for these algorithms were applied to a set of practical problems from a variety of application areas. The methods solved five of these problems in substantially fewer iterations and less time than a standard linear programming approach, but one problem generated an excessive number of nonzero elements. The results seem to indicate that efficiency gains result mainly from eliminating repeated solutions of similar scenario problems.

ACKNOWLEDGMENT

This research was partially supported by the National Science Foundation under Grant ECS-8304065 with The University of Michigan. The author also gratefully acknowledges many helpful comments by the referees.

REFERENCES

ABRAHAMSON, P. G. 1980. A Nested Decomposition Approach for Solving Staircase Structured Linear Programs. In *Proceedings of the Workshop on Large-Scale Linear Programming*, G. Dantzig, M. Dempster, M. Kallio (eds.). IIASA, Laxenburg, Austria, June 2–6.

BEALE, E. M. L. 1955. On Minimizing a Convex Function Subject to Linear Inequalities. *J. Roy. Stat. Soc.* **B17,** 173–184.

BEALE, E. M. L., J. J. H. FORREST AND C. J. TAYLOR. 1980. Multi-time Period Stochastic Programming. In *Stochastic Programming.* M. A. H. Dempster (ed.). Academic Press, New York.

BENDERS, J. F. 1962. Partitioning Procedures for Solving Mixed-Variables Programming Problems. *Numer. Math.* **4,** 238–252.

BIRGE, J. R. 1980. Solution Methods for Stochastic Dynamic Linear Programs. Technical Report SOL 80-29, Systems Optimization Laboratory, Stanford University.

BIRGE, J. R. 1981. Computer Codes for Stochastic Linear Programming. NDST3 and PCST2. Department of Industrial and Operations Engineering, The University of Michigan.

BIRGE, J. R. 1982. The Value of the Stochastic Solution in Stochastic Linear Programs with Fixed Recourse. *Math. Program.* **24,** 314–325.

BRADLEY, S. P., AND D. B. CRANE. 1972. A Dynamic Model for Bond Portfolio Management. *Mgmt. Sci.* **19,** 139–151.

BRADLEY, S. P., AND D. B. CRANE. 1976. *Management of Bank Portfolios.* Wiley, New York.

DANTZIG, G. B. 1955. Linear Programming under Uncertainty. *Mgmt. Sci.* **1,** 197–206.

DANTZIG, G. B., AND A. MADANSKY. 1961. On the Solution of Two-Stage Linear Programs under Uncertainty. In *Proceedings, 4th Berkeley Symposium on Math. Stat. and Prob.* University of California Press, Berkeley.

DANTZIG, G. B., AND P. WOLFE. 1960. Decomposition Principle for Linear Programs. *Opns. Res.* **8,** 101–111.

EVERITT, R., AND W. T. ZIEMBA. 1979. Two-Period Stochastic Programs with Simple Recourse. *Opns. Res.* **27,** 485–502.

FOURER, R. 1979. Solving Staircase L. P.'s by the Simplex Method. Technical Report SOL 79-81, Systems Optimization Laboratory, Stanford University.

FOX, K. A., J. K. SENGUPTA AND E. THORBECKE. 1966. *The Theory of Quantitative Economic Policy.* North-Holland, Amsterdam.

GARSTKA, S., AND D. RUTENBERG. 1973. Computation in Discrete Stochastic Programming with Recourse. *Opns. Res.* **21,** 112–122.

GEOFFRION, A. M. 1970. Elements of Large-Scale Mathematical Programming. *Mgmt. Sci.* **16,** 652–675.

GLASSEY, C. R. 1971. Dynamic Linear Programs for Production Scheduling. *Opns. Res.* **19,** 45–56.

GLASSEY, C. R. 1973. Nested Decomposition and Multistage Linear Programs. *Mgmt. Sci.* **20,** 282–292.

GRINOLD, R. C. 1976. A New Approach to Multi-Stage Stochastic Linear Programs. *Math. Program. Study* **6,** 19–29.

GRINOLD, R. C. 1979. Correction for End Effects in Energy Planning Models. EPRI Report.

GRINOLD, R. C. 1983. Model Building Techniques for the Correction of End Effects in Multistage Convex Programs. *Opns. Res.* **31,** 407–431.

GRINOLD, R. C. 1980. Time Horizons in Energy Planning Models. In *Energy Policy Modeling: United States and Canadian Experiences*, Vol. II, W. T. Ziemba and S. L. Schwartz (eds.). Martinus Nijhoff, Boston.

HO, J. K., AND E. LOUTE. 1981. A Set of Staircase Linear Programming Test Problems. *Math. Program.* **20,** 245–250.

HO, J. K., AND A. S. MANNE. 1974. Nested Decomposition for Dynamic Models. *Math. Program.* **6,** 121–140.

KALL, P. 1979. Computational Methods for Solving Two-Stage Stochastic Linear Programming Problems. *J. Appl. Math. Physics* **30,** 261–271.

KALLBERG, J. G., AND M. I. KUSY. 1976. A Stochastic Linear Program with Simple Recourse. Faculty of Commerce, The University of British Columbia.

KALLBERG, J. G., AND W. T. ZIEMBA. 1981. An Algorithm for Portfolio Revision: Theory, Algorithm Development, and Empirical Results. In *Applications of Management Science*, Vol. I, R. A. Schultz (ed.). JAI Press, Greenwich, Conn.

KALLBERG, J. G., R. W. WHITE AND W. T. ZIEMBA. 1982. Short Term Financial Planning under Uncertainty. *Mgmt. Sci.* **28,** 670–682.

KALLIO, M., AND E. L. PORTEUS. 1977. Decomposition of Arborescent Linear Programs. *Math. Program.* **13,** 348–355.

KUSY, M. I. 1978. An Asset and Liability Management Model. Ph.D. dissertation, Faculty of Commerce, The University of British Columbia.

MURTAGH, B. A., AND M. A. SAUNDERS. 1978. Large Scale Linearly Constrained Optimization. *Math. Program.* **14,** 41–72.

MURTY, K. G. 1968. Two-Stage Linear Programming Under Uncertainty: A Basic Property of the Optimal Solution. *Z. Wahr. Ver. Geb.* **10,** 284–288.

PFEFFERKORN, C. E., AND J. A. TOMLIN. 1976. Design of a Linear Programming System for ILLIAC IV. Technical Report SOL 76-8, Systems Optimization Laboratory, Stanford University.

ROSEN, J. B. 1963. Convex Partition Programming. In *Recent Advances in Mathematical Programming*, R. L. Graves and P. Wolfe (eds.). McGraw-Hill, New York.

STRAZICKY, B. 1980. Some Results Concerning an Algorithm for the Discrete Recourse Problem. In *Stochastic Programming*, M.A.H. Dempster (ed.). Academic Press, New York.

THEIL, H. 1964. *Optimal Decision Rules for Government and Industry.* Rand-McNally, Chicago.

THE UNIVERSITY OF MICHIGAN COMPUTER CENTER. 1980. *MTS*, Vol. **6,** *FORTRAN in MTS.*

VAN SLYKE, R., AND R. WETS. 1969. L-Shaped Linear Programs with Applications to Optimal Control and Stochastic Linear Programs. *SIAM J. Appl. Math.* **17,** 638–663.

WETS, R. 1966. Programming under Uncertainty: the Solution Set. *SIAM J. Appl. Math.* **14,** 1143–1151.

WETS, R. 1975. Solving Stochastic Programs with Simple Recourse, II. In *Proceedings of the Johns Hopkins Symposium on Systems and Information*, pp. 1–6.

WETS, R. 1974. Solving Stochastic Programs with Simple Recourse, I. Department of Mathematics, University of Kentucky.

WETS, R. 1983. Solving Stochastic Programs with Simple Recourse. *Stochastics* **10,** 219–242.