DYNAMIC SPLITTING: AN ALGORITHM FOR DETERMINISTIC AND STOCHASTIC MULTIPERIOD OPTIMIZATION*

DAVID H. SALINGER† AND R. TYRRELL. ROCKAFELLAR‡

Abstract. A new algorithm for the *nonlinear* multistage stochastic programming problem (MSP) is presented; one that is reasonable for the large-scale problem (*e.g.* long term hydropower scheduling) and is highly parallel. The algorithm is based on the application of Spingarn's operator splitting method to the saddle point problem associated with the MSP. The splitting method imposes a decomposability which results in two main subproblems to be solved at each iteration. One is reformulated as an unconstrained linear-quadratic dynamic programming problem and is solved via a linear feedback loop solution extended to the scenario tree structure. The other subproblem is separable into box constrained convex sub-subproblems for each decision state. This crucial separable structure arises only from the splitting of the saddle point problem formulation.

The algorithm was tested on a hydropower scheduling test problem containing 165,000 control variables.

Key words. stochastic programming, dynamic programming, maximal monotone operator, operator splitting, Lagrangian, duality, saddle point, Fenchel duality

1. Introduction.

Dynamic programming is used to model situations where decisions are made in stages. It is well known though that the *dynamic programming algorithm* associated with this modeling framework is ill-suited in general for all but the smallest of these problems. In the new *dynamic splitting* algorithm to be presented here, the nonlinear (convex) multiperiod deterministic or multistage stochastic optimization problem is solved by exploiting a decomposability imposed via an operator splitting. Properly done, this avoids the state-space discretization which dooms the DP algorithm. Our formulation also takes advantage, in one of the resulting subproblems, of the one situation where the DP algorithm shines; the unconstrained linear-quadratic case.

A goal in developing the algorithm was to decompose the multistage stochastic programming problem (MSP) in such a way that the dynamics could be handled easily. The algorithm grew from observation of the special decomposability structure exhibited only by the saddle point formulation of the problem. This underlying modeling structure was laid out by Rockafellar (1998) (see also Salinger, 1997). It will be necessary to review some of that work as a starting point here in Section 2.

Monotone operator splitting methods (see Eckstein (1989) for a thorough review) have long been known in numerical analysis on linear algebra and differential equations. Those operators where predominantly linear and single-valued. Lions and Mercier (1979) extended operator splitting to multivalued monotone operators and applied it to the convex optimization setting. Eckstein and Ferris (1998) have applied operator splitting to the multiperiod deterministic problem with specialization for extended linear-quadratic programs. In the dynamic splitting algorithm presented here (see also Salinger, 1997) operator splitting is, for the first time, applied to multistage stochastic programming. Section 3 reviews some pertinent monotone operator split-

*

†salinger@cbr.washington.edu; Columbia Basin Research, Box 358218, University of Washington, Seattle 98195-8218. This work was partially supported by the Bonneville Power Administration under grant 89-108.

‡rtr@math.washington.edu; Department of Mathematics, Box 354350, University of Washington, Seattle 98195-4350. This work was partially supported by the National Science Foundation under grant DMS-980309

ting results as a starting point for the extension of Spingarn's (1983, 1985) operator splitting method for the saddle point problem.

The new dynamic splitting algorithm is developed in section 4 by applying Spingarn's splitting method to the saddle point formulation of the MSP. The splitting results in two main subproblems to be solved at each iteration. One subproblem is reformulated as an unconstrained dynamic programming problem with quadratic cost and linear dynamics (with scenario tree information/decision structure). This is solved at each iteration via a linear feedback loop solution extended to the scenario tree structure. The other subproblem is separable into box constrained convex minimization problems for each node of the tree. It should be noted that the crucial separable structure of the second subproblem only arises from the appropriate splitting of the saddle point problem formulation associated with the reduced Lagrangian. The resulting algorithmic structure can be easily adapted to parallel computation.

This algorithm is unusual in that it was developed directly for the multistage stochastic programming problem and not as an adaptation of an algorithm for the two-stage stochastic problem or the deterministic problem. It is also unusual in that it is applicable to the nonlinear (convex) problem rather than a linear or piecewise linear objective. Many variations and approximations of the DP algorithm have been tried for these multistage problems. The current fashion for MSP is a stochastic Bender's decomposition approach (cf. Birge (1980, 1985), Wets(1988), Periera and Pinto(1985, 1991)). While the Bender's decomposition theory holds for application of these methods to the nonlinear problem, they have been found to be practical for the linear and piecewise linear problems.

Previous results by Eckstein provide an alternative perspective of the underpinnings of our algorithm. Eckstein and Bertsekas (1992) have shown that the *partial inverse operator* (Spingarn, 1983) is a special case of their *splitting operator* and thus Spingarn's splitting method is a special case of their generalized Douglas-Rachford splitting method (since both splitting methods can be derived by application of the proximal point method (Rockafellar, 1976a) to the *splitting operator*). Eckstein (1994) provides a saddle point application of Douglas-Rachford splitting to convex programming. Also, Eckstein and Ferris (1998) have utilized a generalized Douglas-Rachford splitting to the multiperiod deterministic problem. Our algorithm could then also be viewed as the extension of a saddle point application of Douglas-Rachford splitting to convex multistage stochastic programming. But the Douglas-Rachford iterations (cf. Eckstein, 1989) have a completely different form than the Spingarn iterations we invoke for that splitting. Though Spingarn's scheme can be obtained by invoking Douglas-Rachford for a certain "auxiliary pair" of operators, the two methods do act differently when applied without such a reformulation. That said, we have chosen to present the algorithm as an application of Spingarn's method for reasons of clarity of argument as well as to acknowledge its pedigree.

Section 5 shows an improvement to the algorithm via the introduction of an additional control variable which greatly assists in state constraint compliance. Some performance results from a 165,000 control variable hydropower scheduling test problem will be reported in Section 6. The test problem was constructed from data supplied by the Pacific Gas and Electric Co. in Northern California.

Most of the results presented here are taken from the thesis of Salinger (1997) which contains a more detailed development of the theory as well as the development of a hydropower scheduling model and the application of our algorithm to the scheduling problem.

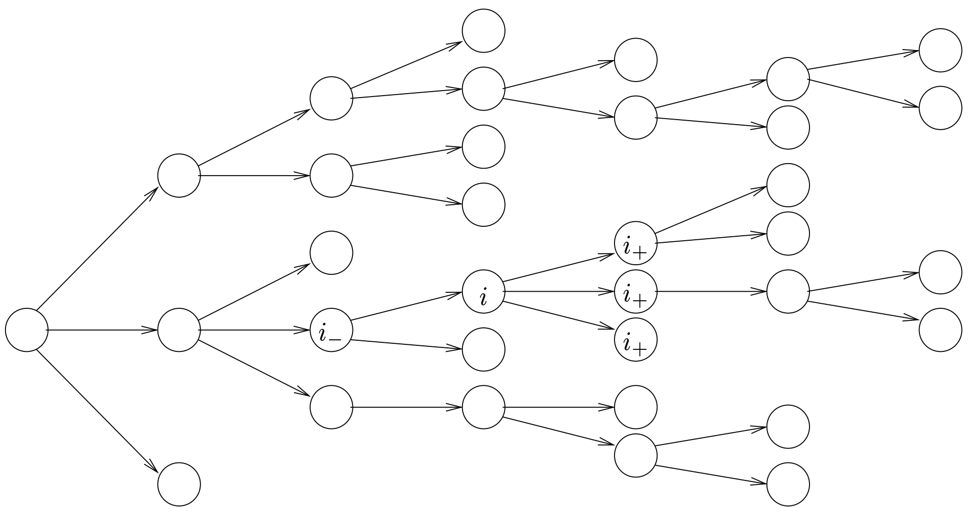


FIG. 2.1. *Scenario tree.*

2. Modeling Structure in Multistage Stochastic Programming. In stochastic optimization, we desire to make an optimal decision (or sequence of decisions) based on information which cannot be known with certainty at the time of the decision. Stochastic programming is characterized by the opportunity, after initial decisions, to make *recourse* decisions in response to additional developments in information. After new information is revealed, a decision is chosen optimally with respect to that information and to expectations on future information. A dynamical constraint (the *dynamics*) is included to track the effects of previous decisions (through the definition of an auxiliary state vector).

Laying out the precise modeling structure is crucial to the development of the dynamic splitting algorithm. This section reviews Rockafellar's (1998) MSP modeling structure in the context of our current purpose (see also Salinger, 1997). The section concludes by specializing saddle point theorems for the convex MSP.

2.1. The Scenario Tree. The underlying information and decision structure of the multistage stochastic model is a scenario tree (see Figure 2.1). The nodes $i \in \mathcal{I}$ of the tree are the information/decision states (also referred to as states or nodes). Node i_- is the unique predecessor of i. The notation i_+ represents the variable index ranging over the successors of i, while $\mathcal{I}_i^+$ represents the set of all successors of i. The set of terminal states (nodes with no following branches) is denoted by $\mathcal{T} \subset \mathcal{I}$.

The transition probability $p_i > 0$ represents the fixed probability of moving from state i_- to state i along a branch of the tree. The *scenario* corresponding to state i is the set of branches leading from the initial state to that state. The scenario probability $\pi_i > 0$ represents the probability of arriving at that state i. It is the product of the transition probabilities along that scenario.

For each information/decision state $i \in \mathcal{I}$, a decision (or control) vector $u_i \in \Re^{n_i}$ is chosen; this defines a *policy mapping* $u : i \mapsto u_i$ for all $i \in \mathcal{I}$. We can also view u as a super-vector of controls in $\Re^N = \Pi_{i \in \mathcal{I}} \Re^{n_i}$.

A particularly attractive feature of the scenario tree structure is the ease in representing the *expected cost*, which we may wish to minimize. The expectation is conditioned on the successor states i_+. If $G_i(u_i, u_{i_-}, \ldots, u_0)$ is the cost incurred at state i due to decision u_i and as a function of all previous decisions in that scenario, then the expected total cost is

$$\sum_{i\in\mathcal{I}} \pi_i G_i(u_i, u_{i_-}, \ldots, u_0).$$

If the G_i are convex in $(u_i, u_{i_-}, \ldots, u_0)$, then the expected cost is a convex function of the policy u.

As well as providing an underlying structure for the decisions, the scenario tree also provides a structure for the information. The scenario tree structure arises from the discrete transition probabilities or a discretization of the continuous transition probabilities. The tree structure makes clear the fact that the probability of reaching a state i_+ is conditioned on the probability of reaching state i in the first place.

There is opportunity to respond (recourse) to developments of information on the scenario tree. That is, future decisions are made after the observation of information. Notice also that the development of information on the tree is not affected by interim decisions.

We will be working directly on the scenario tree, and not on an expanded tree via variable splitting (not to be confused with operator splitting). Advantages of this are that it avoids both the proliferation of variables and the need for nonanticipativity constraints.

2.1.1. Problem Dynamics. As a way of accounting for the effects of past decisions, a state vector $x_i \in \Re^{s_i}$ is defined recursively (by the dynamics) as

$$x_{i_+} = f_{i_+}(x_i, u_i)$$

where $i_+ \in \mathcal{I}_i^+$ represents *any* of the states following state i. Here, we will assume a linear form of the dynamics

$$x_{i_+} = A_{i_+} x_i + B_{i_+} u_i + b_{i_+}$$

where the $b_i \in \Re^{t_i}$ are random variables with discrete probability distribution defined on the scenario tree. The components of b_i, x_i and u_i represent various aspects of the physical system in question at state $i \in \mathcal{I}$.

2.2. The Cost Structure. In general, the physical system will give rise to various constraints. These are incorporated into the cost objective via a penalty (possibly infinite) or multiplier formulation. Borrowing Rockafellar's (1998) notation, write this cost for each $i \in \mathcal{I}$ as

$$F_i(x_i, u_i) = r_i \cdot u_i + \varphi_i(u_i) + \psi_i^*(d_i - C_i x_i - D_i u_i) + c_i \cdot x_i$$

where the extended-real-valued penalty functions φ_i and ψ_i^* are convex, lower semi-continuous (l.s.c.) and proper and where ψ_i^* is the *Fenchel conjugate* (cf. Rockafellar, 1970a) of ψ_i given by

$$\psi_i^*(z_i) = \sup_{v_i \in \Re^{m_i}} \{v_i \cdot z_i - \psi_i(v_i)\}.$$

If for example $\psi_i(v_i) = \delta_{\Re_+^{m_i}}(v_i)$, then the Fenchel conjugate expression is

$$\psi_i^*(d_i - C_i x_i - D_i u_i) = \begin{cases} 0, & \text{if } d_i - C_i x_i - D_i u_i \le 0 \\ +\infty, & \text{otherwise} \end{cases}$$

(where the vector inequality should be interpreted as holding for all components).

This generalized cost formulation melds what have traditionally been called *costs* with what have traditionally been called *constraints*. What is a cost, but a penalty (or soft constraint) on some action?

2.2.1. The General Multistage Stochastic Programming Problem. Collecting the cost and dynamics structures (where G_i is replaced by F_i in conjunction with the dynamics), the multistage stochastic programming problem (MSP) is formulated as follows:

$$\underset{u\in\Re^N}{\text{minimize}}\ f(u)=\sum_{i\in\mathcal{I}}\pi_i F_i(x_i,u_i) \qquad (\mathcal{P}_0)$$

$$\text{where:}\ \ F_i(x_i,u_i)=r_i\cdot u_i+\varphi_i(u_i)+\psi_i^*(d_i-C_ix_i-D_iu_i)+c_i\cdot x_i$$
$$x_{i_+}=A_{i_+}x_i+B_{i_+}u_i+b_{i_+},\quad x_0=b_0$$

where the x_i are viewed as dependent variables defined by the dynamics, where φ_i and ψ_i^* (and ψ_i) are extended-real-valued, convex, l.s.c. and proper, where $\pi_i>0$ is the scenario probability, and where $\Re^N$ is the product of the $\Re^{n_i}$ for all $i\in\mathcal{I}$. For examples of how this structure can exhibit linear, piecewise linear, or extended linear-quadratic programming structure see Rockafellar (1987, 1998) (also Salinger, 1997).

2.2.2. Example. Extended Linear-Quadratic Programming Structure. The special case of extended linear-quadratic programming can be represented in this extended problem $(\mathcal{P}_0)$ format with

$$\varphi_i(u_i)=\delta_{U_i}(u_i)+\frac{1}{2}u_i\cdot R_iu_i,\quad \psi_i(v_i)=\delta_{V_i}(v_i)+\frac{1}{2}v_i\cdot Q_iv_i$$

where U_i,V_i represent polyhedral constraints on u_i and v_i and R_i and Q_i are positive semidefinite. Here, in the minimization, the conjugate function ψ_i^* inflicts piecewise linear-quadratic penalties for violation of $d_i-C_ix_i-D_iu_i\le 0$. (See Rockafellar (1992) for a review of piecewise linear-quadratic penalties.)

2.3. Problem Formulation; The Primal Problem. A constrained convex programming structure can also be represented in this problem $(\mathcal{P}_0)$ format and will be used to provide a primal formulation of the convex multistage stochastic programming problem. Specifically, take for each $i\in\mathcal{I}$

$$\varphi_i(u_i)=\begin{cases} g_i(u_i), & \text{if } u_i\in U_i\\ +\infty, & \text{otherwise}\end{cases}$$

and $\psi_i(v_i)=\delta_{V_i}(v_i)$ where g_i is convex and finite on U_i and the sets U_i and $V_i=[0,\gamma_i]$ (vector interval where $\gamma_i\in\Re_+^{m_i}$) are the product sets of intervals. Then the conjugate expression is

$$\psi_i^*(d_i-C_ix_i-D_iu_i)=\begin{cases} 0, & \text{if } d_i-C_ix_i-D_iu_i\le 0\\ \gamma_i\cdot[d_i-C_ix_i-D_iu_i]_+, & \text{otherwise.}\end{cases}$$

where the notation $[\cdot]_+$ represents the residual (or nonnegative part) of the vector. The components of γ_i specify the level of linear penalty if the corresponding component of the constraint $d_i - C_i x_i - D_i u_i \leq 0$ is violated. By allowing components of γ_i to approach ∞, we get the limiting case of infinite penalties for constraint violation (with the convention that $0 \cdot \infty = 0$). The convex version of the primal problem is then formulated as

$$\underset{u \in \Re^N}{\text{minimize}}\ f(u) = \sum_{i \in \mathcal{I}} \pi_i \left\{ r_i \cdot u_i + \varphi_i(u_i) + \psi_i^*(d_i - C_i x_i - D_i u_i) + c_i \cdot x_i \right\} \qquad (\mathcal{P})$$

$$\text{where:}\quad x_{i_+} = A_{i_+} x_i + B_{i_+} u_i + b_{i_+}, \quad x_0 = b_0,$$

where φ_i and ψ_i^* are given above and where x_i is viewed as a dependent variable.

Notice that the term $\psi_i^*(d_i - C_i x_i - D_i u_i)$, in combination with the dynamics in $(\mathcal{P})$, creates a nonlinear function of u_i, u_{i_-} as well as previous decisions on that scenario. The algorithm to be presented in Section 4 exploits a certain separable structure in i but needs (at highest order) *bilinear* terms in u_i, u_{i_-}. The dual formulation displays similar higher order non-separable terms. But, by formulating the saddle point problem on the reduced Lagrangian, we shall see that this stipulation can be met.

2.3.1. The Reduced Lagrangian. In the formulation of a *full Lagrangian*, the state variables x_i would be viewed as independent variables in addition to u_i. A dual state variable y_i would then be introduced as a multiplier for the dynamical constraint. In the *reduced Lagrangian* formulation, the state variable x_i is interpreted as a dependent variable defined by the dynamics. Then

$$L(u, v) = \sum_{i \in \mathcal{I}} \pi_i \left\{ r_i \cdot u_i + \varphi_i(u_i) + v_i \cdot (d_i - C_i x_i - D_i u_i) - \psi_i(v_i) + c_i \cdot x_i \right\}$$

$$\text{where:}\quad x_{i_+} = A_{i_+} x_i + B_{i_+} u_i + b_{i_+}, \quad x_0 = b_0$$

on $U \times V$ where we use the convention that $\infty - \infty = \infty$.

One should recognize that the reduced Lagrangian was the form used in the development of $(\mathcal{P}_0)$ as well as in the convex version $(\mathcal{P})$.

For completeness, the dual problem can be formulated from the reduced Lagrangian as

$$\underset{v \in \Re^M}{\text{maximize}}\ g(v) = \sum_{i \in \mathcal{I}} \pi_i \left\{ d_i \cdot v_i - \psi_i(v_i) - \varphi_i^* \Big(\big(\sum_{j \in \mathcal{I}_i^+} p_j B_j^\top y_j \big) + D_i^\top v_i - r_i \Big) - b_i \cdot y_i \right\} \qquad (\mathcal{D})$$

$$\begin{aligned} \text{where:}\quad y_i &= \big(\sum_{j \in \mathcal{I}_i^+} p_j A_j^\top y_j \big) + C_i^\top v_i - c_i, &\quad \text{for } i \notin \mathcal{T} \\ y_i - \big[C_i^\top v_i - c_i \big] &= 0, &\quad \text{for } i \in \mathcal{T}. \end{aligned}$$

We now have the primal $(\mathcal{P})$ and dual $(\mathcal{D})$ problem formulations associated with the reduced Lagrangian. Because of its special structure mentioned above, we would also like to formulate the associated saddle point problem.

2.4. Saddle Point Problem Formulation. The saddle point problem for the reduced Lagrangian L on $U \times V$ associated with the convex multistage problem is formulated

$$\begin{aligned} &\text{find } (\overline{u}, \overline{v}) \in U \times V \qquad (\mathcal{S}) \\ &\text{such that :}\ L(\overline{u}, \overline{v}) = \min_{u \in \Re^N} L(u, \overline{v}) = \max_{v \in \Re^M} L(\overline{u}, v) \end{aligned}$$

where the restrictions $u \in U$ and $v \in V$ reside within the indicator functions in L.

If a saddle point exists for L on $U \times V$, the optimal values in $(\mathcal{P})$ and $(\mathcal{D})$ must coincide. Then, the problems $(\mathcal{P})$, $(\mathcal{D})$ and $(\mathcal{S})$ can provide three formulations, each possibly with distinct advantages, for finding $(\overline{u}, \overline{v})$. The algorithm to be presented in this paper relies on a structural advantage found only in the saddle point formulation.

2.4.1. Saddle Point Existence. The following proposition (cf. Rockafellar, 1998) explains the relationship between the solutions of problems $(\mathcal{P})$, $(\mathcal{D})$ and $(\mathcal{S})$ and provides a context for the duality theorems which follow.

PROPOSITION 2.1 (Characterization of Saddle Points).

$$(\overline{u}, \overline{v}) \text{ is a solution to } (\mathcal{S}) \iff \begin{cases} \overline{u} \text{ is a solution to } (\mathcal{P}), \\ \overline{v} \text{ is a solution to } (\mathcal{D}), \\ \inf(\mathcal{P}) = \sup(\mathcal{D}). \end{cases}$$

where $\sup(\mathcal{D})$ *and* $\inf(\mathcal{P})$ *represent the optimal values in problems* $(\mathcal{P})$ *and* $(\mathcal{D})$.

The following theorem, which relates primal optimality for the convex problem $(\mathcal{P})$ and existence of a saddle point of L on $U \times V$ (i.e. a saddle point solution in $(\mathcal{S})$), follows easily from application of theorems in Rockafellar (1970a and 1998) to the current context. (Note: "ri" stands for relative interior.)

THEOREM 2.2 (MSP Duality). *Consider the convex problem $(\mathcal{P})$, the associated reduced Lagrangian L on $U \times V$ and the functions f and g defined in $(\mathcal{P})$ and $(\mathcal{D})$ respectively;*

(a) Suppose that for each $i \in \mathcal{I}$ there exists a $u_i \in \operatorname{ri} U_i$ such that $d_i - C_i x_i - D_i u_i \in \operatorname{ri} \operatorname{dom} \psi_i^*$ where $x_{i_+} = A_{i_+} x_i + B_{i_+} u_i + b_{i_+}$, $x_0 = b_0$. Then $\overline{u}$ is optimal in $(\mathcal{P})$ if and only if there exists $\overline{v}$ such that $(\overline{u}, \overline{v})$ is a saddle point of L on $U \times V$.
or

(a′) Suppose that $-g$ is proper on V and the set $\{v \in V | g(v) \geq \alpha\}$ is bounded for all $\alpha \in \Re$. Then $\overline{u}$ is optimal in $(\mathcal{P})$ if and only if there exists $\overline{v}$ such that $(\overline{u}, \overline{v})$ is a saddle point of L on $U \times V$.

(b) Additionally, if f is proper on U and the set $\{u \in U | f(u) \leq \alpha\}$ is bounded for all $\alpha \in \Re$, then an optimal $\overline{u}$ exists.

Proof. Part (a) follows from (Theorems 6 and 7 in) Rockafellar (1998). Parts (a′) and (b) specialize the duality theorems in (§35 of) Rockafellar (1970a). □

If, for a particular convex MSP, the necessary condition ((a) or (a′)) of Theorem 2.2 is not met, all is not lost. By modifying the MSP to insist that the V_i are all bounded, condition (a′) is automatically met.

The effect of this modification is small. Originally, the V_i were allowed to be unbounded (for example $V_i = \Re_+^{m_i}$). As a result, the $\psi_i^*(d_i - C_i x_i - D_i u_i)$ term in the primal problem would inflict an infinite penalty in the minimization for violation of the constraint $d_i - C_i x_i - D_i u_i \leq 0$. If instead the V_i are bounded, the infinite penalty is replaced by a linear penalty. In practice, this modification will have little effect as the multipliers (penalties) are cranked up until a specified level of constraint compliance occurs. Of course during computation, the penalties never actually reach ∞. If a bound on a component of V_i is found to be too restrictive, it can be increased (in a following run) so that the effect is like the unbounded case.

3. Operator Splitting. Methods of partitioning or decomposition have long been used in mathematics to *reduce* a problem to one of iteratively solving a collection of smaller or easier problems. These types of methods continue to be emphasized with the advent of parallel computing structures which make it possible to to solve the smaller problems all at the same time. One such class of methods is known as *splitting methods for monotone operators* (or operator splitting methods). The main purpose of these methods is to find a zero of a monotone operator (or sum of monotone operators) by "splitting" the operator into two or more simpler operators and then exploiting the special structure of the resulting subproblems.

A monotone operator can be thought of as generalizing, to multivalued nonlinear functions, the property of positive semidefiniteness. The gradient or subgradient of a convex function provides the most famous example of a monotone operator. One can easily see that the minimization of the sum of convex functions becomes the problem of finding a zero of the sum of monotone operators.

This section will provide a way for us to apply Spingarn's operator splitting to the saddle point problem formulation $(\mathcal{S})$ of the multistage stochastic programming problem.

A multivalued function $T:\Re^n \rightrightarrows \Re^n$ is said to be a *monotone operator* if

$$\langle \chi - \chi', \eta - \eta' \rangle \geq 0 \quad \text{whenever} \quad \eta \in T(\chi),\ \eta' \in T(\chi')$$

where $\langle \cdot, \cdot \rangle$ is the inner product. An operator T is said to be *maximal monotone* if it is monotone and its graph

$$G(T) = \{(\chi, \eta) \in \Re^n \times \Re^n | \eta \in T(\chi)\}$$

is not strictly contained in the graph of any other monotone operator $T':\Re^n \rightrightarrows \Re^n$.

3.1. Zero of a Maximal Monotone Operator. Given a maximal monotone operator $T:\Re^n \rightrightarrows \Re^n$, consider the problem of finding a zero of that operator:

$$\text{find } \chi \in \Re^n \text{ such that } 0 \in T(\chi). \tag{$\mathcal{Z}1$}$$

In principle, problem $(\mathcal{Z}1)$ may be solved by such techniques as a *forward* step or *backward* step method.

3.1.1. Operator Splitting. Now imagine that T can be "split" so that $T = T_1 + T_2$ where T_1 and T_2 are also maximal monotone. Our problem is then formulated

$$\text{find } \chi \in \Re^n \text{ such that } 0 \in (T_1 + T_2)(\chi) \tag{$\mathcal{Z}2$}$$

This *operator splitting* can be used to impose a decomposition of the problem into "smaller" problems. We then apply *backward* or *forward* steps to T_1 and T_2 separately in some combination at each iteration.

The Paceman-Rachford and Douglas-Rachford splitting methods (cf. Eckstein, 1989) are well-known especially in the numerical PDE community. Those methods, as well as the forward-backward splitting (cf. Chen and Rockafellar, 1995) employ a combination of forward and backward steps. Another less-known method, due to Spingarn (1983), was derived by applying the proximal point method to a specially contrived *partial inverse* operator associated with the operator $T_1 \times T_2$. This method involves only backward steps. Backward steps exhibit better convergence in general and tend to be less adversely effected by ill-conditioning. (As mentioned previously, Eckstein and Bertsekas (1992) have shown that Spingarn's method can be thought of as a special case of their generalized Douglas-Rachford splitting when a certain auxiliary operator is employed.)

3.2. Spingarn's Operator Splitting Method. Spingarn's (1983) method for finding a zero of the sum of maximal monotone operators (written here for the two-operator case of problem ($\mathcal{Z}2$)) is formulated as follows:

Step 1. Start with arbitrary $\chi \in \Re^n$ and $\eta_1, \eta_2 \in \Re^n$ such that $\eta_1 + \eta_2 = 0$.

Step 2. For $i = 1, 2$, find $\chi_i' \in \Re^n$, $\eta_i' \in \Re^n$ such that

$$\chi + \eta_i = \chi_i' + \eta_i' \quad \text{and} \quad \eta_i' = T_i(\chi_i').$$

Step 3. For $i = 1, 2$, let

$$\chi^+ = \frac{1}{2}(\chi_1' + \chi_2') \quad \text{and} \quad \eta_i^+ = \eta_i' - \frac{1}{2}(\eta_1' + \eta_2').$$

Return to Step 2 with the updated values for χ, η_1 and η_2.

We now derive a more convenient form of Spingarn's algorithm before adapting it for the saddle point problem. To that end, combine the equations in Step 2 to get

$$\begin{aligned} \chi_i' &= (I + T_i)^{-1}(\chi + \eta_i) \\ \eta_i' &= \chi + \eta_i - \chi_i' \end{aligned}$$

for $i = 1, 2$ where the inverse is single valued since the T_i are maximal monotone.

Notice from Step 3 that $\eta_2 = -\eta_1$ and let $\eta = -\eta_1 = \eta_2$. Replace T_i by λT_i ($i = 1, 2$) to introduce a parameter for the purpose of speeding convergence ($0 \in (T_1 + T_2)(\chi) \Leftrightarrow 0 \in (\lambda T_1 + \lambda T_2)(\chi)$). Substituting into Step 3 (and defining iteration index k) gives the following:

THEOREM 3.1 (Spingarn's Method). *An iteration of Spingarn's method for problem ($\mathcal{Z}2$) can be formulated*

$$\begin{aligned} \chi^{k+1} &= \frac{1}{2}\left[(I + \lambda T_1)^{-1}(\chi^k - \eta^k) + (I + \lambda T_2)^{-1}(\chi^k + \eta^k)\right] \\ \eta^{k+1} &= \frac{1}{2}\left[(I + \lambda T_1)^{-1}(\chi^k - \eta^k) - (I + \lambda T_2)^{-1}(\chi^k + \eta^k)\right] + \eta^k \end{aligned} \qquad (\mathcal{SM})$$

where T_1 and T_2 are maximal monotone operators.

The definition of *piecewise polyhedral mapping* is given now as it provides a special case for the convergence theorem which follows.

DEFINITION 3.2 (Piecewise Polyhedral Mapping). *A mapping $M : \Re^n \rightrightarrows \Re^m$ is piecewise polyhedral if its graph is the union of finitely many polyhedral (convex) sets (Rockafellar and Wets, 1998).*

THEOREM 3.3 (Convergence of Spingarn's Method). *Suppose that T_1 and T_2 are maximal monotone. In Spingarn's method ($\mathcal{SM}$) with $\lambda > 0$, if a solution of problem ($\mathcal{Z}2$) exists, then χ^k converges to $\overline{\chi}$ and η^k converges to $\overline{\eta}$ such that $-\overline{\eta} \in T_1(\overline{\chi})$ and $\overline{\eta} \in T_2(\overline{\chi})$ (i.e. $0 \in (T_1 + T_2)(\overline{\chi})$).*

Moreover, if there exists an α such that

$$\|(\chi + \eta) - (\overline{\chi} + \overline{\eta})\| \leq \alpha \|w\| \quad \textit{when} \quad (\chi + \eta) \in (T_1 \times T_2)_A^{-1}(w)$$

for $(w, \chi + \eta)$ close to $(0, \overline{\chi} + \overline{\eta})$, then the convergence of χ^k to $\overline{\chi}$ and η^k to $\overline{\eta}$ is linear and there is an index $\overline{k}$ such that

$$\|(\chi^{k+1} + \eta^{k+1}) - (\overline{\chi} + \overline{\eta})\| \leq \frac{1}{\sqrt{1 + (1/\alpha)^2}} \|(\chi^k + \eta^k) - (\overline{\chi} + \overline{\eta})\| \quad \textit{for } k \geq \overline{k}$$

where $(T_1 \times T_2)_A$ is the partial inverse operator (Spingarn, 1983) associated with operator $(T_1 \times T_2)$.

This extra assumption about the existence of such an α is satisfied in particular if T_1 and T_2 are piecewise polyhedral.

Proof. The first part of the theorem is due to Spingarn (1983). The second part of the theorem is a direct consequence of convergence results for the proximal point method of Rockafellar (1976a) with the less restrictive conditions due to Luque (1984).

If T_1 and T_2 are piecewise polyhedral, then so is $T_1 \times T_2$ as is $(T_1 \times T_2)_A$ since it is a one-to-one linear transformation in the graph space. Then, it is immediate from (Example 9.35 of) Rockafellar and Wets (1998) and the definition of piecewise polyhedral that such an α exists. □

A result about the rate of linear convergence for χ^k (rather than $\chi^k + \eta^k$) with conditions in terms of $T_1 + T_2$ (rather than $(T_1 \times T_2)_A$) might be more desirable. Still, this result implies that if α can be decreased, faster convergence should result.

The exact nature of the relationship between λ and α is not obvious. However, it does seem evident and numerical tests concur, that adjusting λ can help speed convergence.

3.2.1. Application to Minimization. The purpose of this section is to introduce, in the simpler context of minimization, the pattern that will be developed for finding saddle points. In optimization, problem $(\mathcal{Z}1)$ arises from the minimization of a proper convex l.s.c. function h. If $h : \Re^n \to \Re \cup \{+\infty\}$ is convex, then $\partial h : \Re^n \rightrightarrows \Re^n$ is a monotone operator. If h is also l.s.c. and proper, then ∂h is a maximal monotone operator on $\Re^n$ (Rockafellar, 1966). Then the optimality condition $0 \in \partial h(\chi)$ gives rise to problem $(\mathcal{Z}1)$ where $T = \partial h$.

To apply a splitting in the optimization setting, imagine that T_1 and T_2 arise as subgradients of proper convex l.s.c. functions h_1 and h_2 and that we wish to minimize $h = h_1 + h_2$ on $\Re^n$.

$$\underset{u \in \Re^n}{\text{minimize}}\ h(u) = h_1(u) + h_2(u) \tag{$\mathcal{M}$}$$

To formulate algorithm $(\mathcal{SM})$ more conveniently for optimization, we use the fact that when $T = \partial h$ (for h convex l.s.c proper), the backward step $(I + \lambda T)^{-1}(u^k)$ is equivalent to computing a step of the proximal point method:

$$u^{k+1} = \arg \min_{u \in \Re^n} \left\{ h(u) + \frac{1}{2\lambda} \|u - u^k\|^2 \right\}$$

Rockafellar (1976a). Then in $(\mathcal{SM})$, employ the proximal step for the backward steps, let

$$\tilde{u}^k = \chi^k - \eta^k \quad \text{and} \quad \hat{u}^k = \chi^k + \eta^k$$

and define $(u^1_*)_k$ and $(u^2_*)^k$ as the results of the proximal steps in this iteration.

THEOREM 3.4 (Spingarn's Method for Minimization). *An iteration of Spingarn's method for problem $(\mathcal{M})$ can be formulated as follows:*

The proximal steps are

$$(u_*^1)^k = \arg\min_{u\in\Re^n}\left\{h_1(u) + \frac{1}{2\lambda}\|u - \tilde{u}^k\|^2\right\} \qquad (\mathcal{SM}')$$
$$(u_*^2)^k = \arg\min_{u\in\Re^n}\left\{h_2(u) + \frac{1}{2\lambda}\|u - \hat{u}^k\|^2\right\}.$$

Then, the proximal terms are updated as

$$\tilde{u}^{k+1} = (u_*^1)^k + \frac{1}{2}(\tilde{u}^k - \hat{u}^k)$$
$$\hat{u}^{k+1} = (u_*^2)^k + \frac{1}{2}(\hat{u}^k - \tilde{u}^k)$$

and the current iterate toward the solution is

$$u^k = \frac{1}{2}((u_*^1)^k + (u_*^2)^k).$$

The convergence results of Theorem 3.3 hold also for this minimization formulation $(\mathcal{SM}')$ of Spingarn's method where, if h_1 and h_2 are convex l.s.c proper functions on $\Re^n$, then $T_1 = \partial h_1$ and $T_2 = \partial h_2$ are maximal monotone operators. If, in fact, h_1 and h_2 are piecewise quadratic, then T_1 and T_2 are piecewise polyhedral (by Proposition 12.30 of Rockafellar and Wets, 1998). In this case, the convergence is linear.

Notice that in this formulation, u takes the place of χ while the variable η is eliminated. Then, by the convergence of $\eta^k \to \overline{\eta}$ in Theorem 3.3, we get $(u_*^1)^k - (u_*^2)^k \to 0$. With $u^k \to \overline{u}$, this shows that the solution of each proximal subproblem in $(\mathcal{SM}')$ converges to the minimizer (if one exists).

3.2.2. Extension of Spingarn's Method to the Saddle Point Problem. Recall that our purpose is to be able to apply this operator splitting method to the saddle point formulation of the multistage stochastic programming problem. A final step on that path is to extend Spingarn's method to the general saddle point problem. First we review a couple definitions.

DEFINITION 3.5. *Consider any function $L : \Re^n \times \Re^m \to [-\infty, +\infty]$. Then*

(a) $L(u, v)$ is termed a saddle-function *if it is convex in u for all $v \in \Re^m$ and concave in v for all $u \in \Re^n$.*

(b) Let

$$U = \{u \in \Re^n | L(u, v) < +\infty \ \forall v \in \Re^m\}$$
$$V = \{v \in \Re^m | L(u, v) > -\infty \ \forall u \in \Re^n\}.$$

A saddle function L is said to be proper *if the* effective domain *$U \times V$ is non-empty*

DEFINITION 3.6 (Rockafellar, 1970b). *For a saddle function L on $\Re^n \times \Re^m$, the associated operator $T_L(u, v)$ is defined as the set of $(w, z) \in \Re^n \times \Re^m$ such that*

$$\begin{aligned} L(u', v) - \langle u', w\rangle + \langle v, z\rangle &\geq L(u, v) - \langle u, w\rangle + \langle v, z\rangle \qquad (\mathcal{T}_L)\\ &\geq L(u, v') - \langle u, w\rangle + \langle v', z\rangle \end{aligned}$$

for all $u' \in \Re^n$, $v' \in \Re^m$.

The following result, adapted from Rockafellar (1970b), provides conditions for the maximal monotonicity of T_L.

PROPOSITION 3.7. *Let L be a proper saddle-function on $\Re^n \times \Re^m$ such that the function $h_{v'}(u) = L(u, v')$ is lower semicontinuous on $\Re^n$ for all $v' \in \Re^m$ and such that the function $k_{u'}(v) = -L(u', v)$ is upper semicontinuous on $\Re^m$ for all $u' \in \Re^n$. Then T_L is a maximal monotone operator.*

To formulate Spingarn's Method $(\mathcal{SM})$ more conveniently for the saddle point problem $(\mathcal{S})$, we use the fact that for the maximal monotone operator T_L associated with the proper saddle function $L(u, v)$ satisfying the semicontinuity requirements of Proposition 3.7, the backward step $(I + \lambda T_L)^{-1}(u^k, v^k)$ is equivalent to computing a step of the proximal point method as follows (with $\mu = \lambda$):

$$(u^{k+1}, v^{k+1}) = \arg\min_{u\in U}\max_{v\in V}\left\{L(u, v) + \frac{1}{2\lambda}\|u - u^k\|^2 - \frac{1}{2\mu}\|v - v^k\|^2\right\}$$

(Rockafellar, 1976b) where $U\times V$ is the effective domain (introduced in Definition 3.5).

Note that the case where $\mu \neq \lambda$ can be accomodated simply by reconstrueing what norm is used.

Imagine that L can be split so that $L = L_1 + L_2$ with L_1 and L_2 convex-concave and L_1 finite. Then the Spingarn's splitting algorithm can be reformulated for the saddle point problem by applying $(\mathcal{SM})$ to the maximal monotone operators T_{L_1} and T_{L_2} arising from the saddle-functions L_1 and L_2 (with the $\tilde{u}, \hat{u}, \ldots$ notations of the previous section) as follows:

THEOREM 3.8 (Spingarn's Method for the Saddle point Problem). *An iteration of Spingarn's method for problem $(\mathcal{S})$ can be formulated as follows:*
The proximal steps are

$$\begin{aligned}(u_*^1, v_*^1)^k &= \arg\min_{u\in U}\max_{v\in V}\left\{L_1(u, v) + \frac{1}{2\lambda}\|u - \tilde{u}^k\|^2 - \frac{1}{2\mu}\|v - \tilde{v}^k\|^2\right\} \qquad (\mathcal{SM}'')\\ (u_*^2, v_*^2)^k &= \arg\min_{u\in U}\max_{v\in V}\left\{L_2(u, v) + \frac{1}{2\lambda}\|u - \hat{u}^k\|^2 - \frac{1}{2\mu}\|v - \hat{v}^k\|^2\right\},\end{aligned}$$

the proximal term updates are

$$\begin{aligned}(\tilde{u}, \tilde{v})^{k+1} &= (u_*^2, v_*^2)^k + \frac{1}{2}((\tilde{u}, \tilde{v})^k - (\hat{u}, \hat{v})^k)\\ (\hat{u}, \hat{v})^{k+1} &= (u_*^1, v_*^1)^k + \frac{1}{2}((\hat{u}, \hat{v})^k - (\tilde{u}, \tilde{v})^k)\end{aligned}$$

and the current iterates toward the primal and dual solutions are

$$u^k = \frac{1}{2}((u_*^1)^k + (u_*^2)^k) \quad \text{and} \quad v^k = \frac{1}{2}((v_*^1)^k + (v_*^2)^k).$$

The convergence results of Theorem 3.3 hold also for the saddle point version of Spingarn's method $(\mathcal{SM}'')$ where, if L_1 and L_2 are proper saddle functions satisfying the upper and lower semicontinuity requirements of Proposition 3.7, then $T_1 = T_{L_1}$ and $T_2 = T_{L_2}$ (as in Definition 3.6) are maximal monotone operators. In this case, x^k is replaced by $(u, v)^k$. Also, by remarks following Theorem 3.4 the primal and dual solutions of each proximal subproblem in $(\mathcal{SM}'')$ converge to the saddle point (if one exists).

If L_1 and L_2 are linear-quadratic with polyhedral domain $U\times V$, Then T_{L_1} and T_{L_2} are piecewise polyhedral (by Example 12.31 of Rockafellar and Wets, 1998). In this case, the convergence is linear.

4. Application to Multistage Stochastic Programming. This section presents the main result of this paper; the development of the new *dynamic splitting* algorithm for multistage stochastic programming problem $(\mathcal{P})$ introduced in Section 2. The algorithm utilizes Spingarn's method to decompose the saddle point formulation of the problem into two main subproblems to be solved at each iteration. The dynamic subproblem is reformulated as an unconstrained linear-quadratic DP and is solved via a linear feedback loop. Separability of the second subproblem will be exploited, but specific information about function structure is needed to fill in the details of a solution method.

In Section 2, a saddle point problem was formulated relative to the reduced Lagrangian L associated with the convex multistage stochastic problem $(\mathcal{P})$. In Section 3, Spingarn's method was formulated for the saddle point problem $(\mathcal{SM}'')$. The idea here is to apply that method to the multistage stochastic problem with the appropriate choice of splitting $L = L_1 + L_2$ (giving rise to the appropriate operator splitting).

To this end, rewrite the reduced Lagrangian associated with problem $(\mathcal{P})$ as follows:

$$L(u,v) = \underbrace{\sum_{i\in\mathcal{I}} \pi_i \{c_i{\cdot}x_i - v_i{\cdot}C_i x_i\}}_{L_1(u,v)} + \underbrace{\sum_{i\in\mathcal{I}} \pi_i \{r_i{\cdot}u_i + \varphi_i(u_i) + d_i{\cdot}v_i - \psi_i(v_i) + v_i{\cdot}D_i u_i\}}_{L_2(u,v)}$$
$$\text{where: } x_{i_+} = A_{i_+} x_i + B_{i_+} u_i + b_{i_+}, \quad x_0 = b_0,$$

on $U{\times}V$ where $\psi_i(v_i) = \delta_{V_i}(v_i)$ and $\varphi_i(u_i) = g_i(u_i) + \delta_{U_i}(u_i)$ are the convex functions defined in $(\mathcal{P})$. As mentioned previously, we use the convention that $\infty - \infty = \infty$.

This choice of splitting ($L = L_1 + L_2$) results from the desire to exploit the decomposable structure of the reduced Lagrangian. Notice the separable structure of L_2, which can be written

$$L_2(u,v) = \sum_{i\in\mathcal{I}} \pi_i l_i(u_i, v_i)$$
$$\text{where: } l_i(u_i,v_i) = r_i{\cdot}u_i + \varphi_i(u_i) + d_i{\cdot}v_i - \psi_i(v_i) + v_i{\cdot}D_i u_i$$

and is clearly convex-concave in u_i, v_i. This separable structure makes the monotone operator associated with L_2 a strong candidate for the proximal point step.

Notice also that because φ_i and ψ_i can be ∞-valued, L_2 is not differentiable in general. This illustrates the necessity for methods (like proximal point, Spingarn's, forward-backward) which can accommodate *multivalued* monotone operators.

The biaffine structure of

$$L_1(u,v) = \sum_{i\in\mathcal{I}} \pi_i (c_i{\cdot}x_i - v_i{\cdot}C_i x_i)$$
$$\text{where : } x_{i_+} = A_{i_+} x_{i_+} + B_{i_+} u_i + b_{i_+}, \quad x_0 = b_0$$

contains all links between the different decision states. The function L_1 is differentiable, so a forward step for the associated monotone operator might have been considered. But, one of the keys to the algorithm lies in the method for solving the proximal step arising from L_1. General splitting methods that mix forward and backward steps have been discussed, for example, by Chen and Rockafellar (1997).

4.1. The Dynamic Splitting Algorithm. We have chosen the saddle point problem formulation of the multistage problem. We have chosen a particular splitting of the reduced Lagrangian based on its decomposable structure. From the structure of the resulting functions, we have chosen proximal steps for both subproblems and will therefore apply Spingarn's method to the saddle point problem. The true justification for these choices comes in hindsight at the point of having found reasonable methods for the subproblems and later from numerical tests showing that the algorithm actually performs as designed (with the help of assorted implementation tricks).

The dynamic splitting algorithm is derived by applying the saddle point version of Spingarn's method ($\mathcal{SM}''$) to our splitting of the reduced Lagrangian. The proximal subproblems arising at each iteration are then as follows:

$$(u^1_*, v^1_*)^k = \arg\min_{u\in\Re^N}\max_{v\in\Re^M}\sum_{i\in\mathcal{I}}\pi_i\left\{c_i\cdot x_i - v_i\cdot C_i x_i + \frac{1}{2\lambda}\|u_i-\tilde{u}^k_i\|^2 - \frac{1}{2\mu}\|v_i-\tilde{v}^k_i\|^2\right\} \qquad (\mathcal{P}1^k)$$

$$\text{subject to}: x_{i_+} = A_{i_+}x_i + B_{i_+}u_i + b_{i_+}, \quad x_0 = b_0;$$

$$(u^2_*, v^2_*)^k = \arg\min_{u\in\Re^N}\max_{v\in\Re^M}\sum_{i\in\mathcal{I}}\pi_i\left\{l_i(u_i,v_i) + \frac{1}{2\lambda}\|u_i-\hat{u}^k_i\|^2 - \frac{1}{2\mu}\|v_i-\hat{v}^k_i\|^2\right\} \qquad (\mathcal{P}2^k)$$

$$\text{where}: l_i(u_i,v_i) = r_i\cdot u_i + \varphi_i(u_i) + d_i\cdot v_i - \psi_i(v_i) + v_i\cdot D_i u_i.$$

The proximal term updates are

$$(\tilde{u},\tilde{v})^{k+1} = (u^2_*, v^2_*)^k + \frac{1}{2}((\tilde{u},\tilde{v})^k - (\hat{u},\hat{v})^k)$$
$$(\hat{u},\hat{v})^{k+1} = (u^1_*, v^1_*)^k + \frac{1}{2}((\hat{u},\hat{v})^k - (\tilde{u},\tilde{v})^k),$$

and the current iterate toward the primal and dual solutions are

$$u^k = \frac{1}{2}((u^1_*)^k + (u^2_*)^k) \quad\text{and}\quad v^k = \frac{1}{2}((v^1_*)^k + (v^2_*)^k).$$

It should also be noted that at each iteration, the solution of subproblem ($\mathcal{P}1^k$) is feasible in the dynamics while the solution of subproblem ($\mathcal{P}2^k$) is feasible in the control bounds.

THEOREM 4.1 (Convergence Results for Dynamic Splitting). *For the splitting $L = L_1 + L_2$ of problem ($\mathcal{P}$) specified above, the operators T_{L_1} and T_{L_2} are maximal monotone. With $\eta^{k+1} = \frac{1}{2}(u^1_* - u^2_*, v^1_* - v^2_*) - \eta^k$ at each iteration, if a solution of the saddle point problem ($\mathcal{S}$) exists, then (u^k, v^k) converges to $(\overline{u},\overline{v})$ and η^k converges to $\overline{\eta}$ such that $-\overline{\eta}\in T_{L_1}(\overline{u},\overline{v})$ and $\overline{\eta}\in T_{L_2}(\overline{u},\overline{v})$ (*i.e. $0\in(T_{L_1}+T_{L_2})(\overline{u},\overline{v})$*).*

Moreover, if there exists an α such that (with $\chi = (u,v)$)

$$\|(\chi+\eta) - (\overline{\chi}+\overline{\eta})\| \le \alpha\|w\| \quad \textit{when} \quad (\chi+\eta)\in(T_{L_1}\times T_{L_2})^{-1}_A(w)$$

for $(w,\chi+\eta)$ close to $(0,\overline{\chi}+\overline{\eta})$, then the convergence of (u^k,v^k) to $(\overline{u},\overline{v})$ and η^k to $\overline{\eta}$

is linear and there is an index $\overline{k}$ such that

$$\|(\chi^{k+1}+\eta^{k+1}) - (\overline{\chi}+\overline{\eta})\| \le \frac{1}{\sqrt{1+(1/\alpha)^2}}\|(\chi^k+\eta^k) - (\overline{\chi}+\overline{\eta})\| \quad \textit{for } k\ge\overline{k}.$$

where $(T_{L_1} \times T_{L_2})_A$ is the partial inverse operator (Spingarn, 1983) associated with operator $(T_{L_1} \times T_{L_2})$.

This extra assumption about the existence of such an α is satisfied in particular if L_1 and L_2 are linear-quadratic on polyhedral domains.

Proof. This follows directly from Theorem 3.3 and the comments pertaining to linear-quadratic L_1 and L_2 following Theorem 3.8. □

Note, by the convergence of $\eta^k \to \overline{\eta}$, we get $(u_*^1)^k - (u_*^2)^k \to 0$ and $(v_*^1)^k - (v_*^2)^k \to 0$. With $(u^k, v^k) \to (\overline{u}, \overline{v})$, this shows that the solution of each subproblem converges to the saddle point (if one exists).

As in the remarks made after Theorem 3.3, a result about the rate of linear convergence for $\chi^k = (u^k, v^k)$ (rather than $\chi^k + \eta^k$) with conditions in terms of $T_{L_1} + T_{L_2}$ (rather than $(T_{L_1} \times T_{L_2})_A$) might be more desirable. Still, this result implies that if α can be decreased, faster convergence should result.

The exact nature of the relationship between λ and α is not obvious. However, it does seem evident and numerical tests concur, that adjusting λ can help speed convergence.

It remains to be explained how to solve the subproblems.

4.2. Subproblem $(\mathcal{P}1^k)$: The Dynamic Subproblem. At each iteration, subproblem $(\mathcal{P}1^k)$ is a saddle point problem in its own right and therefore has an associated primal problem of the form

$$\underset{u\in\Re^N}{\text{minimize}}\ f_1^k(u) = \sup_{v\in\Re^M} \left\{ \sum_{i\in\mathcal{I}} \pi_i (c_i \cdot x_i - v_i \cdot C_i x_i + \frac{1}{2\lambda}\|u_i - \tilde{u}_i^k\|^2 - \frac{1}{2\mu}\|v_i - \tilde{v}_i^k\|^2) \right\}$$
$$\text{where:}\quad x_{i_+} = A_{i_+} x_i + B_{i_+} u_i + b_{i_+}, \quad x_0 = b_0$$

Noticing that the supremand is concave and unconstrained in v_i for all $i \in \mathcal{I}$, we set the gradient to zero in order to determine the max.. Solving that equation reveals that

$$\overline{v}_i = \tilde{v}_i^k - \mu C_i x_i, \quad \forall\, i \in \mathcal{I}.$$

Substituting gives

$$f_1^k(u) = \sum_{i\in\mathcal{I}} \pi_i \left\{ (c_i - C_i \tilde{v}_i^k)\cdot x_i + \frac{1}{2}\mu (C_i x_i)\cdot C_i x_i + \frac{1}{2\lambda}\|u_i - \tilde{u}_i^k\|^2 \right\}.$$

We desire to write the resulting minimization in a general form. To this end, let $Q_i = \mu C_i^\top C_i$, $q_i = c_i - C_i \tilde{u}_i^k$, $R_i = \frac{1}{\lambda} I$, and $r_i = -\frac{1}{\lambda}\tilde{u}_i^k$. Dropping constant terms (and with the assumption that no decision is made at the terminal states $i \in \mathcal{T}$), the minimization becomes

$$\underset{u\in\Re^M}{\text{minimize}} \sum_{i\in\mathcal{I}/\mathcal{T}} \pi_i \left\{ \frac{1}{2} x_i \cdot Q_i x_i + q_i \cdot x_i + \frac{1}{2} u_i \cdot R_i u_i + r_i \cdot u_i \right\}$$
$$+ \sum_{i\in\mathcal{T}} \pi_i \left\{ \frac{1}{2} x_i \cdot Q_i x_i + q_i \cdot x_i \right\} \qquad (\mathcal{DP}_{lq})$$
$$\text{where: } x_{i_+} = A_{i_+} x_i + B_{i_+} u_i + b_{i_+}, \quad x_0 = b_0.$$

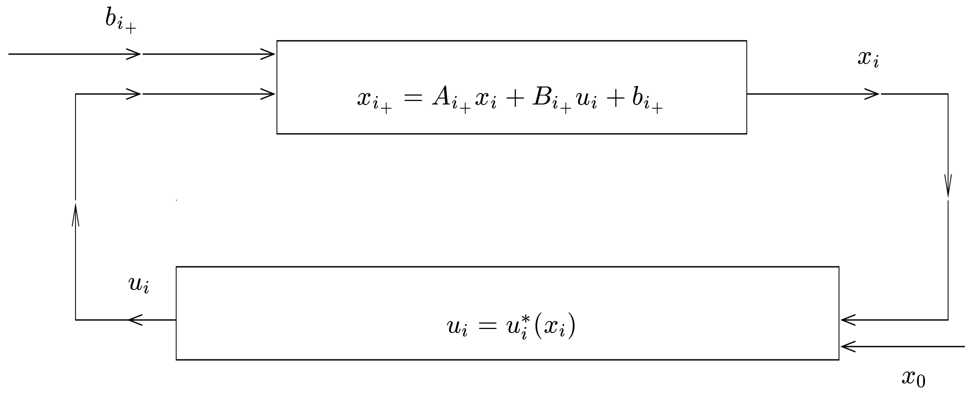


FIG. 4.1. *Linear feedback loop solution for the unconstrained linear-quadratic DP on the scenario tree (where $u_i^*(x_i)$ is defined in Theorem 4.2).*

Formulated in this way, the resulting problem can be thought of as an unconstrained dynamic programming problem on the scenario tree with linear dynamics and quadratic cost; though we would instead think of the problem as one of choosing a minimizing *policy* $\{u_i(x_i)\}_{i\in\mathcal{I}/\mathcal{T}}$ rather than a minimizing supervector $\{u_i\}_{i\in\mathcal{I}/\mathcal{T}}$.

Rockafellar has shown that if $\{u_i^*(x_i)\}_{i\in\mathcal{I}/\mathcal{T}}$ is an optimal policy for the dynamic programming version of the problem, then $\{\overline{u}_i\}_{i\in\mathcal{I}/\mathcal{T}}$ (where $\overline{u}_i = u_i^*(x_i^*)$, $x_{i_+}^* = A_{i_+}x_i^* + B_{i_+}\overline{u}_i + b_{i_+}$, $x_0^* = x_0$ for $i \in \mathcal{I}/\mathcal{T}$) is an optimal solution for the MSP version of the problem (see Salinger, 1997).

This special case of unconstrained dynamic programming with quadratic cost and linear dynamics is well known to be one of the few general cases where a closed form ("linear feedback loop") solution can readily be determined. The following theorem extends that celebrated result (see, for instance, Bertsekas, 1987) to the DP on a scenario tree structure. The proof is an easy induction argument.

THEOREM 4.2 (Feedback Loop Solution for $(\mathcal{DP}_{lq})$). *A linear feedback loop solution (see Figure 4.1) to problem $(\mathcal{DP}_{lq})$ can be obtained for each $i \in \mathcal{I}/\mathcal{T}$ as*

$$u_i^*(x_i) = -\big(R_i + \sum_{j\in\mathcal{I}_i^+} p_j B_j^\top K_j B_j\big)^{-1}\big(r_i + \sum_{j\in\mathcal{I}_i^+} p_j B_j^\top (l_j + K_j(b_j + A_j x_i))\big)$$

where x_i is given by the dynamics and where the symmetric positive semidefinite matrix K_i and the vector l_i is defined, for $i \in \mathcal{T}$, as $K_i = Q_i$ and $l_i = q_i$ and recursively defined, for $i \notin \mathcal{T}$, as

$$\begin{aligned} K_i = Q_i &+ \sum_{j\in\mathcal{I}_i^+} p_j A_j^\top K_j A_j \\ &- \Big(\sum_{j\in\mathcal{I}_i^+} p_j A_j^\top K_j B_j\Big)\Big(R_i + \sum_{j\in\mathcal{I}_i^+} p_j B_j^\top K_j B_j\Big)^{-1}\Big(\sum_{j\in\mathcal{I}_i^+} p_j B_j^\top K_j A_j\Big) \\ l_i = q_i &+ \sum_{j\in\mathcal{I}_i^+} p_j A_j^\top (l_j + K_j b_j) \\ &- \Big(\sum_{j\in\mathcal{I}_i^+} p_j A_j^\top K_j B_j\Big)\Big(R_i + \sum_{j\in\mathcal{I}_i^+} p_j B_j^\top K_j B_j\Big)^{-1}\Big(r_i + \sum_{j\in\mathcal{I}_i^+} p_j B_j^\top (l_j + K_j b_j)\Big). \end{aligned}$$

4.2.1. Practical Concerns. To employ this DP algorithm for the subproblem, first calculate the K_i, l_i starting with $i \in \mathcal{T}$ and working in a single "backward" pass of the scenario tree. Then calculate the u_i^* in a single "forward" pass (see Figure 4.1) while updating x_i in the dynamics.

The matrix inversion $(R_i + \sum_{j \in \mathcal{I}_i^+} p_j B_j^\top K_j B_j)^{-1}$ does not cause any diffuculty in the solution process. Notice first that since R_i is positive definite and Q_i is positive semidefinite, then K_i is also positive semidefinite and $R_i + \sum_{j \in \mathcal{I}_i^+} p_j B_j^\top K_j B_j$ is positive definite. Therefore the inverse exists.

Secondly, K_i, R_i, B_i do not change from one iteration to the next if λ and μ are fixed (as they should be). Therefore, the inverse can be calculated for each $i \in \mathcal{I}$ in the initial iteration of the algorithm and used throughout. (Matlab can invert a 1000 by 1000 matrix in under a second, so this is not a bottleneck when it is expected that the algorithm will need on the order of 500 iterations.)

Thirdly, notice in our formulation of the subproblem that R_i did not depend on i. The *incidence* matrix B_i is often independent of i. If the C_i (and therefore Q_i) also do not depend on i, or depend on i only to the extent that within each decision stage (*e.g.* month, etc.) the Q_i are identical, then the K_i within each stage are also identical. In this case (which was the situation for the hydropower scheduling problem) the matrix inverse $(R_i + \sum_{j \in \mathcal{I}_i^+} p_j B_j^\top K_j B_j)^{-1}$ needs only to be calculated once for each decision stage and only in the first iteration of the solution process.

One final simplification to the dynamic subproblem can be made if the matrices A_i and B_i are block diagonal (as is exhibited in a hydropower scheduling problem with two or more distinct watersheds). In this case, the subproblem is separable by diagonal block (or by watershed in the hydro example).

4.3. Subproblem ($\mathcal{P}2^k$); The Separable Subproblem. Utilizing the separability of L_2 and the norms, we can formulate subproblem ($\mathcal{P}2^k$) as follows:
For each $i \in \mathcal{I}$

$$(u_{*i}^2, v_{*i}^2)^k = \arg \min_{u_i \in \Re^{n_i}} \max_{v_i \in \Re^{m_i}} \left\{ l_i(u_i, v_i) + \frac{1}{2\lambda} \|u_i - \hat{u}_i^k\|^2 - \frac{1}{2\mu} \|v_i - \hat{v}_i^k\|^2 \right\}$$
$$\text{where: } l_i(u_i, v_i) = r_i \cdot u_i + \varphi_i(u_i) + d_i \cdot v_i - \psi_i(v_i) + v_i \cdot D_i u_i.$$

Recall that $\varphi_i(u_i)$ takes a value of $+\infty$ for $u_i \notin U_i$ and that $\psi_i(v_i) = \delta_{V_i}(v_i)$. Thus, the subproblem can be formulated for each $i \in \mathcal{I}$ as

$$(u_{*i}^2, v_{*i}^2)^k = \arg \min_{u_i \in U_i} \max_{v_i \in V_i} \Big\{ r_i \cdot u_i + g_i(u_i) + d_i \cdot v_i + v_i \cdot D_i u_i + \frac{1}{2\lambda} \|u_i - \hat{u}_i^k\|^2 - \frac{1}{2\mu} \|v_i - \hat{v}_i^k\|^2 \Big\}$$

where U_i and V_i represent box constraints on u_i and v_i.

Further, if v_i is unconstrained (alternatively if u_i is unconstrained) then this saddle point problem should be written in equivalent primal (alt. dual) form. Then the supremum (alt. infimum) problem would be unconstrained in v_i (alt. u_i) and could be solved in closed form (as was done with the supremum in subproblem ($\mathcal{P}1^k$)).

Without knowledge of the exact nature of g_i as well as U_i and V_i, it would be difficult to be specific about how to continue from here. An example of this for the long term hydropower scheduling problem can be found in Salinger (1997) and will be published later in a companion paper. There, the subproblem is further separated into easy quadratic subproblems for the dual, primal "spilled release" and additional control (defined in Section 5) variables. The primal "turbined release" variable subproblem, where g_i is formed from the sum over subperiods of the integral of marginal cost data over power shortage in that subperiod, is solved using properties of Fenchel duality.

5. An Improvement. This section presents a practical improvement which was seen to greatly improve constraint compliance without changing the overall problem structure. Return to our splitting of the reduced Lagrangian on page 13 and notice that some of the terms of the constraint equation $d_i - C_i x_i - D_i u_i \leq 0$ occur in L_1 and some in L_2. When the resulting subproblems are solved, this splitting up of the constraint terms results in neither subproblem aiming for compliance of this constraint. Although Theorem 4.1 promises linear convergence under this splitting, as a practical matter this constraint was seemingly ignored by the algorithm in the course of test problem computation.

If all terms associated with that constraint were to be grouped in L_2 for the purposes of the splitting, the separability of subproblem $(\mathcal{P}2^k)$ would be ruined. If all terms associated with that constraint were to be grouped in L_1 for the purposes of the splitting (including ψ_i), the unconstrained nature of the dynamic subproblem $(\mathcal{P}1^k)$ would be ruined. Another fix is needed.

To combat this constraint compliance problem, we modify the reduced Lagrangian via an additional control variable.

5.1. The Modified Reduced Lagrangian. As a step toward formulating a modified reduced Lagrangian for problem $(\mathcal{P})$, an additional control variable $w_i \in \Re^{m_i}$ is introduced along with the stipulation that

$$w_i - x_i = 0.$$

The terms

$$v_i \cdot (d_i - C_i x_i - D_i u_i) - \psi_i(v_i)$$

of the reduced Lagrangian, possibly corresponding in the associated primal problem to the state constraint

$$d_i - C_i x_i - D_i u_i \leq 0,$$

are then replaced by

$$v_i \cdot (d_i - C_i w_i - D_i u_i) - \psi_i(v_i) + v_i' \cdot (w_i - x_i)$$

where the multiplier $v_i' \in \Re^{m_i}$ is introduced for the new constraint $w_i - x_i = 0$.

Then a modified reduced Lagrangian associated with problem $(\mathcal{P})$ can be formulated

$$L(u, w; v, v') = \sum_{i \in \mathcal{I}} \pi_i \left\{ r_i \cdot u_i + \varphi_i(u_i) + v_i \cdot (d_i - C_i w_i - D_i u_i) - \psi_i(v_i) + v_i' \cdot (w_i - x_i) + c_i \cdot x_i \right\}$$

$$\text{where: } x_{i_+} = A_{i_+} x_i + B_{i_+} u_i + b_{i_+}, \quad x_0 = b_0$$

on $(U \times \Re^M) \times (V \times \Re^M)$ where $\psi_i(v_i) = \delta_{V_i}(v_i)$ and $\varphi_i(u_i) = g_i(u_i) + \delta_{U_i}(u_i)$ are the convex functions defined in $(\mathcal{P})$.

5.2. The Splitting. Then, the modified reduced Lagrangian is split as follows to arrive at an operator splitting exhibiting the same structure as before but now with a mechanism for constraint compliance within subproblem $(\mathcal{P}2^k)$:

$$\begin{aligned} L_1(u,w;v,v') &= \sum_{i\in\mathcal{I}} \pi_i \left\{ c_i\cdot x_i + v_i'\cdot(w_i - x_i) \right\} \\ &\text{where: } x_{i_+} = A_{i_+}x_i + B_{i_+}u_i + b_{i_+}, \quad x_0 = b_0 \\ L_2(u,w;v,v') &= \sum_{i\in\mathcal{I}} \pi_i \left\{ r_i\cdot u_i + \varphi_i(u_i) + v_i\cdot(d_i - C_iw_i - D_iu_i) - \psi_i(v_i) \right\} \end{aligned}$$

on $(U\times\Re^M)\times(V\times\Re^M)$ where φ_i and ψ_i are defined above.

Notice that the structure exhibited here is identical to the structure of the original formulation (with (u_i, w_i) replacing u_i and (v_i, v_i') replacing v_i and with the proper resizing of B_i, D_i etc. Thus, all previous theorems and the subproblem solution methods remain valid for this modified formulation.

6. Performance Results. There are various criteria for measuring algorithm performance including CPU times and solution accuracy measurements. A standard measure of accuracy of the solution is the *duality gap* which is reported along with the norm of constraint violations. This information can also provide useful stopping criteria for the algorithm.

The algorithm was implemented, with various "implementation tricks," for a large long-term hydropower scheduling problem constructed from data supplied by the Pacific Gas and Electric Co. The sparsely branched scenario tree (supplied by PG&E) consisted of 27 scenarios spanning 24 months. There were 165,200 control variables representing turbined or spilled releases of water (350 control decisions at each of 472 decision nodes) and 44,368 state variables representing water storage. (For more on the test problem and implementation, see Salinger (1997).)

The test problem and algorithm were prototyped in Matlab. Running on a DEC Alphastation 500/333 (333MHz EV5 processor), the algorithm used about one minute of CPU time per iteration. But, Matlab is notoriously slow for large problems with large loops. If the algorithm were to be implemented using a "faster" language and by taking advantage of the ample opportunity for parallelization, it seems certain that reasonable run times would result.

Results of the duality gap measurements and the norm of constraint violations are presented in Figures 6.1 – 6.3. Figure 6.1 is a plot of the primal and dual objective values at each iteration as well as the true cost $(\sum_{i\in\mathcal{I}} \pi_i g_i(u_i))$. Due to the above mentioned implementation tricks, and to better reflect the actual problems being solved at each iteration, the primal and dual objectives contain some penalty (multiplier) terms. The reported value of these objectives is likely inflated due in part to small infeasibilities in the test problem.

Reporting of the duality gap is further complicated by the fact that the primal and dual solutions at each iteration are not necessarily feasible. Recall that in each iteration, the solution of the dynamic subproblem (P1k) is always feasible in the dynamics and that the solution of the separable subproblem (P2k) is always feasible in the control constraints. But, as the reported *solution* at each iteration is the mean of the two subproblem solutions, it need not be feasible in the dynamics nor the controls. In fact, there are times early in the solution process when the sign of the multipliers (dual variables) is clearly out of step with the constraints. As a result, it appears that we are reporting a negative duality gap in that iteration. With enough

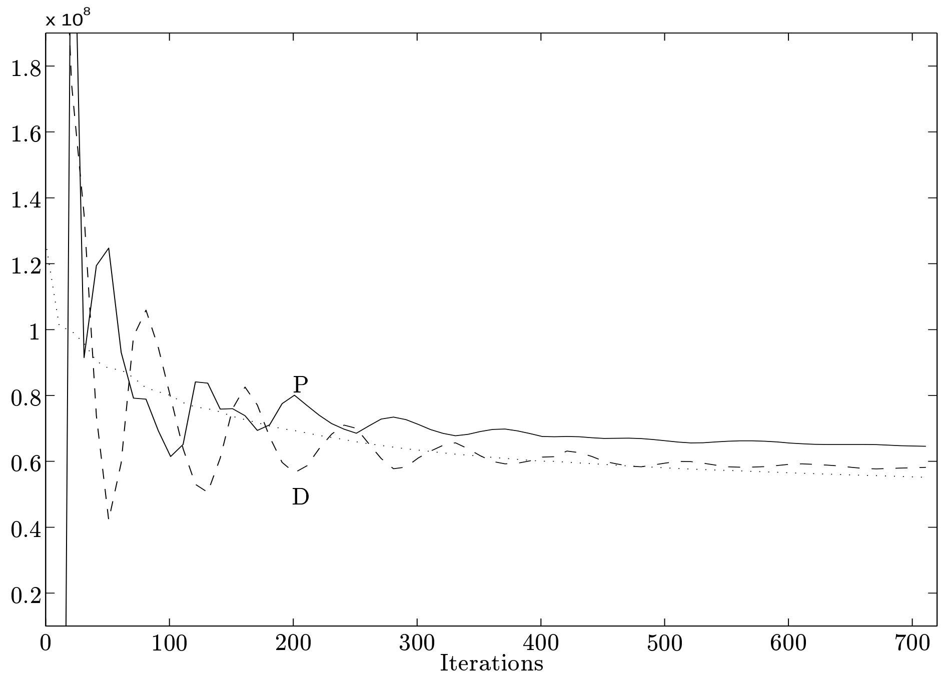


FIG. 6.1. *The value of the primal (P) and dual (D) objectives and the true cost (dotted line) at each iteration. (Note: the primal objective contains some penalty information in addition to the true cost.)*

iterations, the approximate solution at each iteration becomes more nearly feasible, and so the reported duality gap better approximates the actual duality gap.

The duality gap is usually reported as a ratio of

$$\frac{[\text{Primal Objective}] - [\text{Dual Objective}]}{[\text{Primal Objective}]}$$

A plot of this ratio at each iteration is shown in Figure 6.2. The fairly large gap (0.1) is likely due in part to small infeasibilities in the test problem.

Figures 6.3 (a) and (b) show the norm of the expected value of the state constraint violations and the norm of the control constraint $u_i \in \mathcal{I}$ violations. Recall that the solution of the dynamic subproblem is always feasible in the dynamics while the solution of the separable subproblem is always feasible in the box constraints. Recall also that the current solution at each iteration is the average of the solutions of the two subproblems at that iteration. Thus, Figure 6.3(b) also gives an idea of the violation of the dynamical constraint.

REFERENCES

[1] D. P. Bertsekas, *Dynamic Programming*, Prentice-Hall, inc., Englewood Cliffs, N.J., 1987.

[2] J. R. Birge, *Solution methods for stochastic dynamic linear programs*, tech. report, Report 80/29, Systems Optimization Lab.,Dept of Operations Research, Stanford University, CA, 1980.

[3] ———, *Decomposition and partitioning methods for multistage stochastic linear programs*, Operations Research, 33(5) (1985), pp. 989–1007.

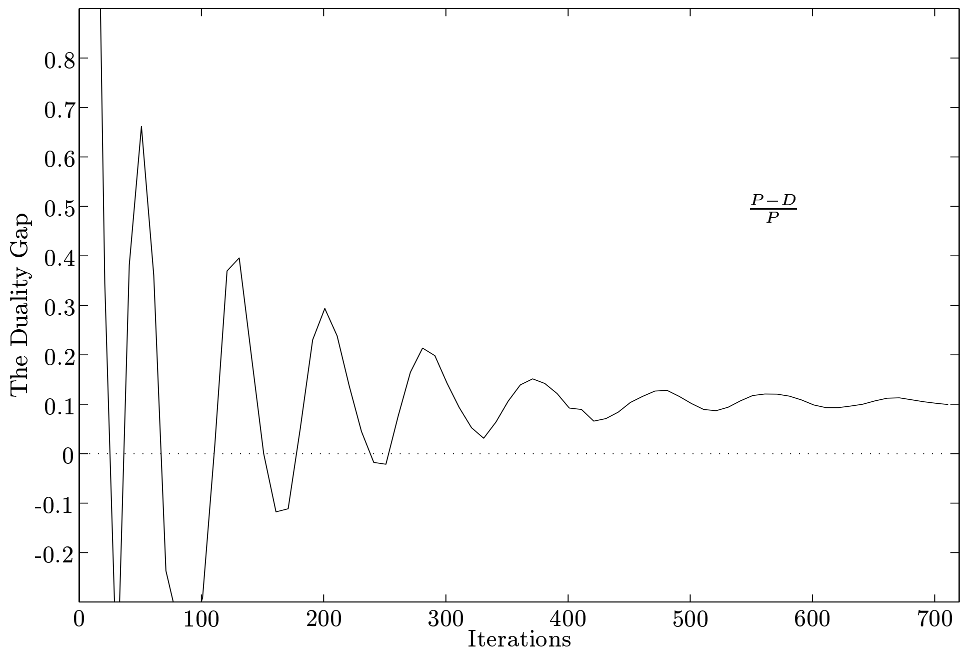


FIG. 6.2. *The Duality gap at each iteration.*

[4] G. H.-G. CHEN AND R. T. ROCKAFELLAR, *Convergence rates in forward-backward splitting*, Siam J. Optimization, 7 (1997), pp. 421–444.

[5] J. ECKSTEIN, *Splitting methods for monotone operators with applications to parallel optimization*, PhD thesis, Massachusetts Institute of Technology, 1989.

[6] ———, *Some saddle point splitting methods for convex programming*, Optimization Methods and Software, 4 (1994), pp. 75–83.

[7] J. ECKSTEIN AND D. P. BERTSEKAS, *On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators*, Math. Programming, 55 (1992), pp. 203–243.

[8] J. ECKSTEIN AND M. C. FERRIS, *Operator splitting methods for monotone affine variational inequalities, with a parallel application to optimal control*, INFORMS J. on Computing, 10 (1998), pp. 218–235.

[9] P. L. LIONS AND B. MERCIER, *Splitting algorithms for the sum of two nonlinear operators*, SIAM J. Numer Anal., 16 (1979), pp. 964–979.

[10] F. J. LUQUE, *Asymptotic convergence analysis of the proximal point algorithm*, SIAM J. Control and Opt., 22 (1984), pp. 277–293.

[11] M. V. F. PEREIRA AND L. M. V. G. PINTO, *Stochastic optimization of a multireservoir hydroelectric power system*, Water Resour. Res., 21(6) (1985), pp. 779–792.

[12] ———, *Stochastic dual dynamic programming*, Mathematical Programming, 52 (1991), pp. 359–375.

[13] R. T. ROCKAFELLAR, *Characterization of subdifferentials of convex functions*, Pacific Journal of Mathematics, 17(3) (1966), pp. 497–510.

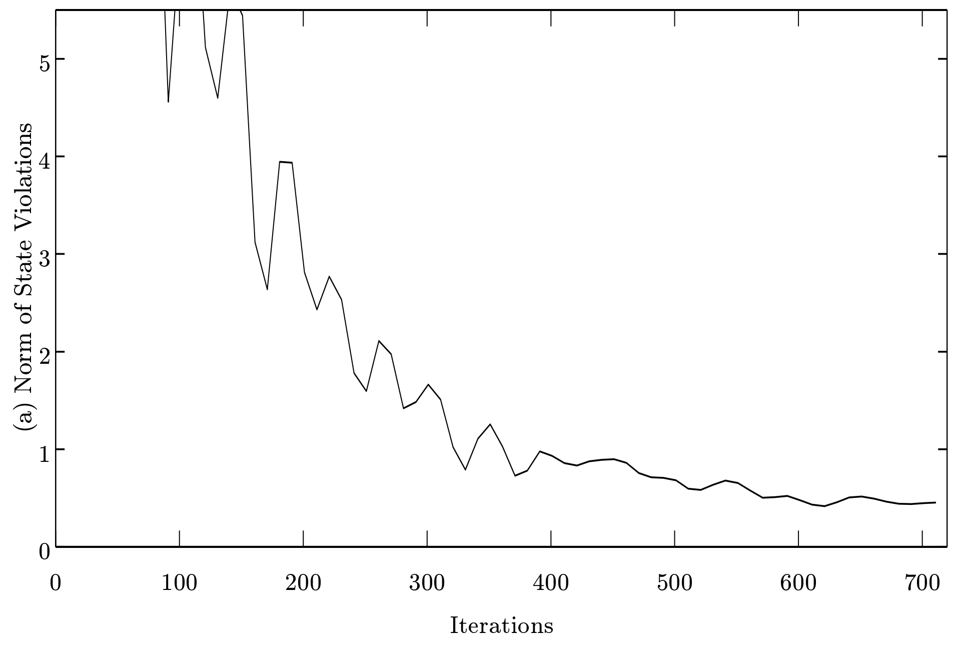
[14] ———, *Convex Analysis*, Princeton University Press, Princeton N.J., 1970a.

[15] ———, *Monotone operators associated with saddle-functions and minimax problem*, in Proceedings of Symposia in Pure Mathematics, F. Browder, ed., vol. 18(1), American Mathematical Society, 1970b, pp. 241–250.

[16] ———, *Monotone operators and the proximal point algorithm*, SIAM J. Control And Optimization, 14 (1976a), pp. 877–898.

[17] ———, *Augmented lagrangians and applications of the proximal point algorithm*, Math. Oper. Res., 1 (1976b), pp. 97–116.

[18] ———, *Linear-quadratic programming and optimal control*, SIAM J. Control and Optimization,



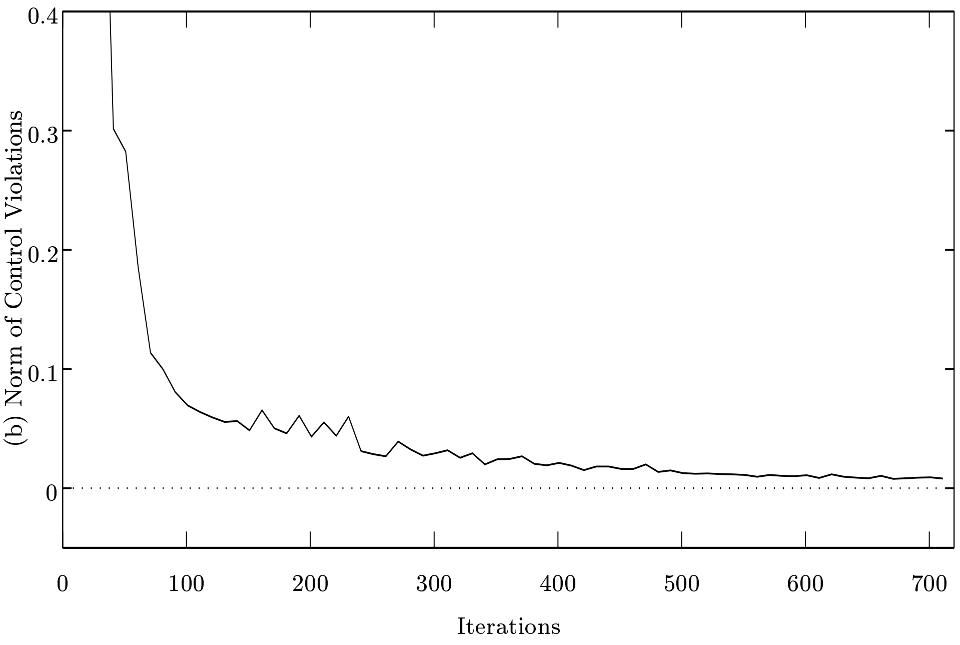


FIG. 6.3. *(a) The 2-norm of the state-constraint violation at each iteration. (b) The 2-norm of the primal control-constraint violation at each iteration.*

25 (1987), pp. 781–814.

[19] ———, *Extended linear-quadratic programming*, SIAM J. Optimization, News and Views, 1 (1992), pp. 3–6.

[20] ———, *Lagrange multipliers and optimality*, SIAM Review, 35 (1993), pp. 183–236.

[21] ———, *Duality and optimality in multistage stochastic programming*, Annals of Operations Research, December (1998).

[22] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*, Springer-Verlag, 1998.

[23] D. H. Salinger, *A splitting algorithm for multistage stochastic programming with application to hydropower scheduling*, PhD thesis, Dept of Applied Math., University of Washington, 1997.

[24] J. E. Spingarn, *Partial inverse of a monotone operator*, Appl. Math. Optim., 10 (1983), pp. 247–265.

[25] ———, *Applications of the method of partial inverses to convex programming: decomposition*, Mathematical Programming, 32 (1985), pp. 199–223.

[26] R. J.-B. Wets, *Large scale linear programming techniques*, in Numerical Results for Stochastic Optimization, Y. Ermoliev and R. Wets, eds., Springer, Berlin, 1988, ch. 3.