Relaxations of Approximate Linear Programs for the Real Option Management of Commodity Storage

Selvaprabu Nadarajah, François Margot, Nicola Secomandi
Tepper School of Business, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213-3890, USA
{snadaraj, fmargot, ns7}@andrew.cmu.edu

Tepper Working Paper 2011-E5
February 2012; Revised: May 2012

Abstract

The real option management of commodity conversion assets gives rise to intractable Markov decision processes (MDPs). This is due primarily to the high dimensionality of a commodity forward curve, which is part of the MDP state when using high dimensional models of the evolution of this curve, as commonly done in practice. Focusing on commodity storage, we develop a novel approximate dynamic programming methodology that hinges on the relaxation of approximate linear programs (ALPs) obtained using value function approximations based on reducing the number of futures prices that are part of the MDP state. We derive equivalent approximate dynamic programs (ADPs) for a class of these ALPs, also subsuming a known ADP. We obtain two new ADPs, the value functions of which induce feasible policies for the original MDP, and lower and upper bounds, estimated via Monte Carlo simulation, on the value of an optimal policy of this MDP. We investigate the performance of our ADPs on existing natural gas instances and new crude oil instances. Our approach has potential relevance for the approximate solution of MDPs that arise in the real option management of other commodity conversion assets, as well as the valuation and management of real and financial options that depend on forward curve dynamics.

1 Introduction

Real options are models of projects that exhibit managerial flexibility (Dixit and Pindyck 1994, Trigeorgis 1996). In commodity settings, this flexibility arises from the ability to adapt the operating policy of commodity conversion assets to the uncertain evolution of commodity prices. For example, consider a merchant that manages a natural gas storage asset (Maragos 2002). This merchant can purchase natural gas from the wholesale market at a given price, and store it for future resale into this market at a higher price. Other examples of commodity conversion assets include assets that produce, transport, ship, and procure energy sources, agricultural products, and metals.

Managing commodity conversion assets as real options (Smith and McCardle 1999, Geman 2005) gives rise to, generally, intractable Markov Decision Processes (MDPs). In a given stage, the state of such an MDP includes both endogenous and exogenous information. The endogenous information describes the current operating condition of the conversion asset, while the exogenous information represents current market conditions. Changes in the endogenous information are caused by managerial decisions that modify the asset operating condition. In contrast, the exogenous information evolves as a result of market dynamics. The MDP intractability is due primarily to the common use in practice of high dimensional models of the evolution of the exogenous information (Eydeland and Wolyniec 2003, Gray and Khandelwal 2004). To illustrate, consider the MDP for the real option management of a commodity storage asset formulated by Lai et al. (2010) using a multi-maturity version of the Black (1976) model of futures price evolution. The endogenous information in this MDP is the asset available inventory at a given date, a one dimensional variable; the exogenous information in this MDP is the commodity forward curve at a given time, an object with much higher dimensionality than inventory.

Approximations are thus typically needed to solve such MDPs. These approximations involve determining a feasible policy, and estimating both its value, which yields a lower bound on the value of an optimal policy, and an upper bound on the value of an optimal policy. In this paper we focus on the approximate solution of the intractable commodity storage MDP formulated by Lai et al. (2010; LMS for short). To address this intractability, LMS propose an Approximate Dynamic Program (ADP) based on a value function that in each stage only depends on the spot price, in addition to the inventory level, and ignores all the other elements of the forward curve. Applied to natural gas instances, their model computes near optimal policies, provided it is sequentially reoptimized, and fairly tight dual upper bounds (Glasserman 2004, Chapter 8, and Brown et al. 2010). This Storage ADP (SADP) features a peculiar conditional expectation that makes it solvable. It is however unclear whether this expectation might serve some other purpose.

The investigation of this conditional expectation is the starting point of our analysis. We show that SADP is a relaxation of a math program that is equivalent to an Approximate Linear Program (ALP; Schweitzer and Seidmann 1985, de Farias and Van Roy 2003) obtained from their MDP. The stated conditional expectation in SADP enacts this relaxation. This relaxation is useful because it alleviates the negative consequences, which we identify, of formulating an ALP using a value function approximation that ignores a subset of the forward curve.

We leverage these insights by developing a novel approximate dynamic programming methodology that we name Partitioned Surrogate Relaxation (PSR). Our PSR approach hinges on the relaxation of ALPs obtained from the commodity storage MDP using value function approximations that ignore a subset of the elements of the forward curve, thus reducing the dimensionality of the exogenous information in the state of this MDP. Given a partition of the ALP constraint set, we replace each set in this partition by a surrogate constraint obtained as a positive linear combination of the constraints in this set using predefined multipliers.

Our approach subsumes SADP since SADP is only one of the approximate models that can be obtained by applying our methodology. We also obtain two new ADPs: one based on a value function approximation that in each stage depends on the spot price and the inventory level, and one that also depends on variables the price of the prompt month futures contract, that is, the one with delivery in the next stage. These ADPs satisfy more general conditions that ensure that a PSR relaxation of an ALP or of an equivalent math programming reformulation thereof yields an ADP.

The value functions of our ADPs induce feasible policies for the original MDP, also leveraging ADP reoptimization. Monte Carlo simulation of such policies yields estimates of valid greedy lower bounds on the value of an optimal policy of this MDP. We also use Monte Carlo simulation to estimate valid dual upper bounds on the value of these policies.

We benchmark the bounds computed by our ADPs on the LMS natural gas instances and a newly created set of crude oil instances. Our reoptimized lower bounds are near optimal both on the natural gas and crude oil instances. In particular, they are comparable to the LMS reoptimized lower bounds on the natural gas instances. Our upper bounds either match or improve on the LMS upper bounds for natural gas, and are essentially tight for crude oil. Compared to SADP, one of our ADPs has a substantial computational advantage, and similar lower and upper bounding performance; our other ADP has a smaller computational requirement without reoptimization and delivers stronger upper bounds, but has a larger computational burden with reoptimization (however this ADP does not rely as much on reoptimization as SADP and ADP1 do to obtain competitive lower bounds).

Although our focus is on commodity storage, our proposed methodology has potential relevance for the approximate solution of intractable MDPs that arise in the real option management of other commodity conversion assets, as well as the valuation and management of real and financial options (see the discussion in Secomandi et al. 2011, §1 for examples) that depend on forward curve dynamics; that is, MDPs whose state includes both endogenous and exogenous information.

The remainder of this paper is organized as follows. We review the extant literature in §2. We provide background material in §3. In §4, we analyze SADP. We present our PSR method and our two new ADPs in §5. In §6, we analyze the optimal value functions of these two ADPs and their associated bounds, focusing on a tractable version of the commodity storage MDP. We discuss the computational complexity of a specification of our approach in §7. We present our numerical results in §8. We conclude in §9.

2 Literature Review

Approximate dynamic programming has received substantial attention in the recent literature. Bertsekas and Tsitsiklis (1996), Van Roy (2002), Adelman (2006), Chang et al. (2007), and Powell (2011) are excellent sources on this topic. Schweitzer and Seidmann (1985) introduce the approximate linear programming approach to approximate dynamic programming. de Farias and Van Roy (2001, 2003, 2006) analyze it. Applications of this approach include Trick and Zin (1997) in economics; Adelman (2004) and Adelman and Klabjan (2011) in inventory control; Adelman (2007), Farias and Van Roy (2007), and Zhang and Adelman (2009) in revenue management; and Morrison and Kumar (1999), de Farias and Van Roy (2001, 2003), Moallemi et al. (2008), and Veatch (2010) in queuing. The novelty of our work relative to this literature is two fold. The first is the presence of exogenous information in the state of the commodity storage MDP that we consider, whereas this type of information is absent in most of the models studied in the extant approximate linear programming literature. The second is our development and use of the PSR approach to deal with the difficulties brought about by using a lower dimensional representation of this information in an ALP. Our approach relies on a novel application of surrogate relaxation (Glover 1968, 1975) in an approximate linear programming context.

The use of constraint relaxations in approximate linear programming is relatively new and the literature is scant. Petrik and Zilberstein (2009) and Desai et al. (2011) use constraint relaxation to improve the value function approximation obtained by solving an ALP: Petrik and Zilberstein (2009) propose a relaxation method for ALPs that penalizes violated constraints in the objective function; the method of Desai et al. (2011) relaxes an ALP by allowing budgeted violation of constraints. Our surrogate relaxation approach is different from the ones used by these authors.

As in LMS, we use the information relaxation and duality approach for upper bound estimation discussed by Brown et al. (2010), which generalizes earlier work by Rogers (2002), Andersen and Broadie (2004), and Haugh and Kogan (2004). However, our approach is more general than the one of LMS. We also introduce new ADPs, adding to the literature on commodity storage valuation (e.g., Chen and Forsyth 2007, Boogert and De Jong 2008, Thompson et al. 2009, Carmona and Ludkovski 2010, Secomandi 2010, Wu et al. 2010, Birge 2011, Boogert and De Jong 2011, Secomandi et al. 2011, and Felix and Weber 2012). More broadly, our PSR approach potentially provides a solution methodology for other real option problems. Our approach differs from least squares Monte Carlo methods (Longstaff and Schwartz 2001, Tsitsiklis and Van Roy 2001), which could be used to solve such problems (Cortazar et al. 2008), because it is based on linear programming rather than regression.

3 Background Material

In §§3.1-3.2 we present the commodity storage MDP and the bounding approach that we use. These subsections are in part based on §2 and §4.2 in LMS. In §3.3 we discretize the state and action spaces of this MDP.

3.1 Commodity Storage MDP

A commodity storage asset provides a merchant with the option to purchase and inject, store, and withdraw and sell a commodity during a predetermined finite time horizon, while respecting injection and withdrawal capacity limits, as well as inventory constraints. The merchant's goal is to maximize the market value of the storage asset. We model this valuation problem as an MDP. Purchases and injections and withdrawals and sales give rise to cash flows. The storage asset has N possible dates with cash flows. The i-th cash flow occurs at time T_i, $i \in \mathcal{I} := \{0, \ldots, N-1\}$. Each such time is also the maturity of a futures contract. We thus focus on determining the value of storage due to futures, rather than spot, price volatility; that is, monthly, rather than, daily volatility. We denote as $F_{i,j}$ the futures price at time T_i of a contract maturing at time T_j, $j \geq i$. The forward curve is the collection of futures prices $F_i := (F_{i,j}, i \in \mathcal{I}, j \geq i)$. We adopt the convention $F_N \equiv 0$. We also define $F'_i := (F_{i,j}, i \in \mathcal{I}, j > i)$, $\forall i \in \mathcal{I}$, for notational convenience.

The set of feasible inventory levels is $\mathcal{X} := [0, \bar{x}]$, where 0 and $\bar{x} \in \mathbb{R}_+$ represent the minimum and maximum inventory levels, respectively. The absolute value of the injection capacity C^I (< 0) and the withdrawal capacity C^W (> 0) represent the maximum amounts that can be injected and withdrawn in between two successive futures contract maturities, respectively. An action a corresponds to an inventory change during this time period. A positive action represents a withdrawal and sell decision, a negative action a purchase and inject decision, and the zero action is the do nothing decision. Define $\cdot \wedge \cdot \equiv \min\{\cdot,\cdot\}$ and $\cdot \vee \cdot \equiv \max\{\cdot,\cdot\}$. The set of feasible injections, withdrawals, and overall actions are $\mathcal{A}^I(x) := \left[C^I \vee (x - \bar{x}), 0\right]$, $\mathcal{A}^W(x) := \left[0, x \wedge C^W\right]$, and $\mathcal{A}(x) := \mathcal{A}^I(x) \cup \mathcal{A}^W(x)$, respectively.

The immediate reward from taking action a at time T_i is the function $r(a, s_i)$, where $s_i \equiv F_{i,i}$ is the spot price at this time. The coefficients $\alpha^W \in (0, 1]$ and $\alpha^I \geq 1$ model commodity losses associated with withdrawals and injections, respectively. The coefficients c^W and c^I represent withdrawal and injection marginal costs, respectively. The immediate reward function is defined as

$$r(a,s) := \begin{cases} (\alpha^I s + c^I)a, & \text{if } a \in \mathbb{R}_-, \\ 0, & \text{if } a = 0, \qquad \forall s \in \mathbb{R}_+, \\ (\alpha^W s - c^W)a, & \text{if } a \in \mathbb{R}_+. \end{cases} \tag{1}$$

Let Π denote the set of all the feasible storage policies. Given the initial state (x_0, F_0), valuing a storage asset entails finding a policy in this set that realizes the maximum time 0 market value $V_0(x_0, F_0)$ of this asset in this state. Thus, we are interested in solving the following problem:

$$V_0(x_0, F_0) := \max_{\pi \in \Pi} \sum_{i \in \mathcal{I}} \delta^i \mathbb{E}\left[r(a_i^\pi(\tilde{x}_i^\pi, \tilde{F}_i), \tilde{s}_i) | x_0, F_0\right], \tag{2}$$

where δ is the risk free discount factor from time T_i back to time T_{i-1}, $\forall i \in \mathcal{I} \setminus \{0\}$; $\mathbb{E}$ is expectation under the risk neutral measure for the forward curve evolution (this measure is unique in our setting); x_i^π is the inventory level realized at time T_i when using policy π; and $a_i^\pi(x_i, F_i)$ is the action taken by policy π at time T_i in state (x_i, F_i). Problem (2) can be equivalently formulated as the following commodity storage MDP, which we refer to as the Exact Dynamic Program (EDP):

$$V_N(x_N, F_N) := 0, \quad \forall x_N \in \mathcal{X}, \tag{3}$$

$$V_i(x_i, F_i) = \max_{a \in \mathcal{A}(x_i)} r(a, s_i) + \delta \mathbb{E}\left[V_{i+1}\left(x_i - a, \tilde{F}_{i+1}\right) | F_i'\right], \quad \forall i \in \mathcal{I}, (x_i, F_i) \in \mathcal{X} \times \mathbb{R}_+^{N-i}, \tag{4}$$

where $V_i(x_i, F_i)$ is the optimal value function in stage i and state (x_i, F_i), and we assume that F_i' is sufficient to compute the expectation.

Consistent with the practice-based literature (Eydeland and Wolyniec 2003, Chapter 5, Gray and Khandelwal 2004, and the discussion in LMS), we assume that EDP is formulated using a full dimensional model of the risk neutral evolution of the forward curve. An example is the multi-maturity version of the Black (1976) model of futures price evolution, which we use for our computational experiments. In this model, the time t futures price with maturity at time T_i, $F(t, T_i)$, evolves as a driftless Brownian motion with maturity specific and constant volatility $\sigma_i > 0$. The instantaneous correlation between the standard Brownian motion increments $dZ_i(t)$ and $dZ_j(t)$ corresponding to the futures prices with maturities T_i and T_j, $i \neq j$, is $\rho_{ij} \in (-1, 1)$ $(\rho_{ii} = 1)$. This model is

$$\frac{dF(t, T_i)}{F(t, T_i)} = \sigma_i dZ_i(t), \ \forall\, i \in \mathcal{I}, \tag{5}$$

$$dZ_i(t) dZ_j(t) = \rho_{ij} dt, \ \forall\, i, j \in \mathcal{I}, \, i \neq j. \tag{6}$$

Model (5)-(6) can be extended by making the constant volatilities and instantaneous correlations time dependent. This would not affect our analysis in §§4-6.

Proposition 3.1, based on Secomandi et al. (2011, Proposition 4 and Lemma 2), provides structural properties of the optimal value function and an optimal policy of EDP. These properties serve as a reference for comparing the structural properties of the ADPs discussed in §5.

Proposition 3.1. *(a) In every stage $i \in \mathcal{I}$, the value function $V_i(x_i, F_i)$ is concave in $x_i \in \mathcal{X}$ for each given $F_i \in \mathbb{R}_+^{N-i}$; and (b) an optimal policy for EDP features two base-stock targets, $\underline{b}_i(F_i)$ and $\bar{b}_i(F_i) \in \mathcal{X}$, which depend on i and F_i; these targets are such that $\underline{b}_i(F_i) \leq \bar{b}_i(F_i)$ and an optimal* *action $a_i^*(x_i, F_i)$ satisfies*

$$a_i^*(x_i, F_i) = \begin{cases} C^I \vee [x_i - \underline{b}_i(F_i)], & \text{if } x_i \in [0, \underline{b}_i(F_i)), \\ 0, & \text{if } x_i \in [\underline{b}_i(F_i), \bar{b}_i(F_i)], \\ C^W \wedge [x_i - \bar{b}_i(F_i)], & \text{if } x_i \in (\bar{b}_i(F_i), \bar{x}]. \end{cases} \tag{7}$$

Moreover, if C^I, C^W, and $\bar{x}$ are integer multiples of some maximal number $Q \in \mathbb{R}_+$, then (c) $V_i(x_i, F_i)$ is piecewise linear and continuous in $x_i \in \mathcal{X}$ for each $F_i \in \mathbb{R}_+^{N-i}$; (d) $V_i(\cdot, F_i)$ can change slope only at integer multiples of Q; and (e) $\underline{b}_i(F_i)$ and $\bar{b}_i(F_i)$ can be chosen to be integer multiples of Q.

3.2 Bounding Approach

In general, computing an optimal policy for EDP under a price model such as (5)-(6) is computationally intractable (see §6 for an exception). We now describe a procedure based on Monte Carlo simulation for estimating lower and upper bounds on the EDP optimal value function in the initial stage and state, as well as obtaining a feasible policy for EDP, given an approximation to the EDP value function. We illustrate this procedure using the value function approximation $\hat{V}_i(x_i, s_i)$, which we assume is available. This function only uses the spot price s_i from the forward curve F_i. Nevertheless, the same approach extends to value function approximations that depend on a larger subset of prices in this forward curve.

Consider lower bound estimation. Given an inventory level x_i and a forward curve F_i in stage i, we use $\hat{V}_i(x_i, s_i)$ as an approximation of $V_i(x_i, F_i)$ to compute a feasible action in stage i and state (x_i, F_i). We do this by solving the greedy optimization problem

$$\max_{a \in \mathcal{A}(x_i)} r(a, s_i) + \delta \mathbb{E}\left[\hat{V}_{i+1}(x_i - a, \tilde{s}_{i+1}) \, | F_{i,i+1}\right], \tag{8}$$

where we assume that $F_{i,i+1}$ is sufficient for computing the expectation; for example, this is the case with the price model (5)-(6). In computations, we numerically approximate this expectation, e.g., as explained in §7. We obtain (8) from (4) by replacing $V_{i+1}(\cdot,\cdot)$ with $\hat{V}_{i+1}(\cdot,\cdot)$ and F_i' with $F_{i,i+1}$. We apply the action $a_i(x_i, s_i)$ computed in (8), which we assume is unique and refer to as the *greedy* action, and sample the forward curve F_{i+1} to obtain the new state $(x - a_i(x_i, s_i), F_{i+1})$. We continue in this fashion until we reach time T_{N-1}. We then discount back to time T_0 and cumulate the values of the cash flows generated by this process starting from the given state (x_0, F_0) at stage 0. We repeat this process over multiple samples, each time starting from the state (x_0, F_0) at time 0, and average the sample discounted total cash flows to estimate the value of the greedy policy, that is, the policy defined by the greedy actions in each stage and state. This provides us with an estimate of a greedy lower bound on the EDP value of storage, $V_0(x_0, F_0)$.

When a value function approximation is computed by an ADP, as discussed in §§4-5, it is typically possible to generate an improved greedy lower bound estimate by sequentially reoptimizing this ADP to update its value function approximations within the Monte Carlo simulation used for lower bound estimation. Specifically, solving an ADP at time T_i yields value function approximations for stages i through $N-1$. However, we only implement the greedy action induced by the stage i value function approximation. At time T_{i+1}, we re-optimize the "residual" ADP, that is, the one defined over the remaining stages $i+1$ through $N-1$, given the inventory level resulting from performing this action and the newly available forward curve. We repeat this procedure until time T_{N-1}. Repeating this process over multiple price samples allows us to estimate a reoptimized greedy lower bound.

For upper bound estimation, we use the information relaxation and duality approach for MDPs (see Brown et al. 2010, and references therein). We sample a sequence of spot price and prompt month futures price pairs $P_0 := ((s_i, F_{i,i+1}))_{i=0}^{N-1}$ starting from the forward curve F_0 at time 0. We use our value function approximation $\hat{V}(x_i, s_i)$ to define the following dual penalty for executing the feasible action a in stage i and state (x_i, F_i) given knowledge of the prompt month futures price $F_{i,i+1}$ and the spot price in stage $i+1$, s_{i+1}:

$$p_i(x_i, a, s_{i+1}, F_{i,i+1}) := \hat{V}_{i+1}(x_i - a, s_{i+1}) - \mathbb{E}\left[\hat{V}_{i+1}(x_i - a, \tilde{s}_{i+1}) | F_{i,i+1}\right]. \quad (9)$$

For computational purposes, we numerically approximate this expectation, e.g. as discussed in §7. This penalty approximates the value of knowing the next stage spot price when performing this action. Then, we solve the following deterministic dynamic program given the sequence P_0:

$$U_i(x_i; P_0) = \max_{a \in \mathcal{A}(x)} r(a, s_i) - p_i(x_i, a, s_{i+1}, F_{i,i+1}) + \delta U_{i+1}(x_i - a; P_0), \quad (10)$$

$\forall i \in \mathcal{I}$ and $x_i \in \mathcal{X}$, with boundary condition $U_N(x_N; P_0) := 0$, $\forall x_N \in \mathcal{X}$. In (10), the per stage reward $r(a, s_i)$ is modified by the penalty $p_i(x_i, a, s_{i+1}, F_{i,i+1})$ for using the future information available in P_0. We solve a collection of deterministic dynamic programs specified by (10), each one corresponding to a sample sequence P_0. We estimate a dual upper bound denoted by $U_0(x_0, F_0)$ on the EDP value of storage in stage 0 and state (x_0, F_0), $V_0(x_0, F_0)$, as the average of the value functions of these deterministic dynamic programs in this stage and state; that is, we compute an estimate of $U_0(x_0, F_0) := \mathbb{E}\left[U_0(x_0; \tilde{P}_0)|F_0\right]$, where the expectation is taken with respect to the risk neutral distribution of the random sequence $\tilde{P}_0$. This estimate can be obtained efficiently when the maximization in (10) can be reduced to an optimization over a finite set of actions. This is the case with the value function approximations that we develop in this paper, as discussed in §§5.2-5.3.

3.3 Discretized Commodity Storage MDP

EDP has continuous state and action spaces in every stage. Our analysis in the rest of this paper relies on formulating a discretized version of EDP, labeled as DDP, as an equivalent linear program (Puterman 1994, §6.9). We now introduce DDP.

Under the assumption in Proposition 3.1, which holds in the remainder of this paper, we can optimally discretize the continuous inventory set $\mathcal{X}$ into the finite set $\mathcal{X}^D := \{0, Q, 2Q, \ldots, \bar{x}\}$, and the feasible action set $\mathcal{A}(x)$ for inventory level $x \in \mathcal{X}^D$ into the finite set

$$\mathcal{A}^D(x) := \{\left[C^I \vee (x - \bar{x})\right], \left[C^I \vee (x - \bar{x})\right] + Q, \left[C^I \vee (x - \bar{x})\right] + 2Q, \ldots, \left[x \wedge C^W\right]\}.$$

We let $\mathcal{F}_i^D \subset \mathbb{R}_+^{N-i}$ represent a finite set of forward curves at time T_i, and denote by $\mathcal{F}_{i,j}^D \subset \mathbb{R}_+$ the finite set of values of the futures price $F_{i,j}$ when this price belongs to the forward curve $F_i \in \mathcal{F}_i^D$. We also suppose that each set $\mathcal{F}_i^D$ is available. In addition, assume that we have available a joint probability mass function defined on $\mathcal{F}_{i+1}^D$ for the random vector $(\tilde{s}_{i+1}, \tilde{F}_{i+1,i+2}, \ldots, \tilde{F}_{i+1,N-1})$ conditional on the futures price vector $(F_{i,i+1}, F_{i,i+2}, \ldots, F_{i,N-1}) \in \mathcal{F}_i^D$. For instance, such discretized sets and associated probability mass functions could be obtained using lattice techniques, as discussed in §7.

Replacing the continuous sets that define EDP with the discretized sets discussed in this subsection yields DDP:

$$V_N^D(x_N, F_N) := 0, \quad \forall x_N \in \mathcal{X}^D, \tag{11}$$

$$V_i^D(x_i, F_i) = \max_{a \in \mathcal{A}^D(x_i)} r(a, s_i) + \delta \mathbb{E}\left[V_{i+1}^D\left(x_i - a, \tilde{F}_{i+1}\right)|F_i'\right], \quad \forall i \in \mathcal{I}, (x_i, F_i) \in \mathcal{X}^D \times \mathcal{F}_i^D, \tag{12}$$

where $V_i^D(x_i, F_i)$ is the DDP optimal value function in stage i and state (x_i, F_i), and the expectation is expressed with respect to the probability mass function discussed in the previous paragraph. The optimal value functions and an optimal policy of DDP satisfy properties equivalent to the ones stated in Proposition 3.1.

In the rest of this paper, we assume that the futures price vector $(F_{i,i+1}, F_{i,i+2}, \ldots, F_{i,i+j})$ is sufficient to obtain the joint probability mass function of the random vector $(\tilde{s}_{i+1}, \tilde{F}_{i+1,i+2}, \ldots, \tilde{F}_{i+1,i+j})$ for $j = 1, \ldots, N-1$. In particular, this implies that F_i' is the only information required to determine the joint probability mass function of the random forward curve $\tilde{F}_{i+1}$. This assumption is satisfied by the multi-maturity Black model (5)-(6).

4 Analysis of SADP

In this section, we use math programming to analyze SADP, that is, the ADP model of LMS. This analysis yields two key insights that set the stage for the development of our PSR methodology in §5.

Denote by $\phi_i(x_i, s_i)$ an approximation of the DDP value function, $V_i^D(x_i, F_i)$, in stage i and state (x_i, F_i). This value function approximation depends on the inventory $x_i \in \mathcal{X}_i^D$ and the spot price $s_i \in \mathcal{F}_{i,i}^D$. SADP in our notation is

$$\textbf{SADP:} \quad \phi_i(x_i, s_i) = \mathbb{E}\left[\max_{a \in \mathcal{A}(x_i)} r(a, s_i) + \delta \mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1})|\tilde{F}_{i,i+1}\right] |s_i, F_{0,i+1}\right], \tag{13}$$

$\forall i \in \mathcal{I}$ and $(x_i, s_i) \in \mathcal{X}^D \times \mathcal{F}_{i,i}^D$, with $\phi_N(x_N, s_N) := 0$, $\forall x_N \in \mathcal{X}^D$. The maximization in (13) is analogous to the maximization in (12) but uses $\phi_i(\cdot, s_{i+1})$ in lieu of $V_{i+1}(\cdot, F_{i+1})$. The maximization in (13) depends on the inventory level x_i, the spot price s_i, and the random futures price $\tilde{F}_{i,i+1}$, while the value function approximation in the left hand side of (13) is a function of only x_i and s_i. Therefore, the first expectation term in (13), that is, $\mathbb{E}\left[\cdot|s_i, F_{0,i+1}\right]$, makes the value function approximation $\phi_i(x_i, s_i)$ computable. Our analysis in this section sheds additional light on the role played by this expectation.

To analyze SADP, we formulate the following math program, which we label the Storage Math Program (SMP):

$$\textbf{SMP:} \quad \min_{\phi} \sum_{i \in \mathcal{I}, x_i \in \mathcal{X}^D, s_i \in \mathcal{F}^D_{i,i}} \phi_i(x_i, s_i) \tag{14}$$

$$\text{s.t. } \phi_i(x_i, s_i) \geq \mathbb{E}\left[\max_{a \in \mathcal{A}(x_i)} r(a, s_i) + \delta \mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1}) | \tilde{F}_{i,i+1}\right] | s_i, F_{0,i+1}\right],$$

$$\forall i \in \mathcal{I}, (x_i, s_i) \in \mathcal{X}^D \times \mathcal{F}^D_{i,i}, \tag{15}$$

$$\phi_N(x_N, s_N) = 0, \forall x_N \in \mathcal{X}^D. \tag{16}$$

The SMP decision variables are the terms $\phi_i(x_i, s_i)$, which are constrained by (15) and (16). SMP is analogous to the equivalent linear programming version of an MDP (Puterman 1994, §6.9). Proposition 4.1 states that solving SMP is equivalent to solving SADP.

Proposition 4.1. *An optimal solution to SMP solves SADP.*

Proof. There is a single constraint (15) for each triple (i, x_i, s_i). We claim that this constraint holds as an equality in an optimal solution to SMP. Fix an optimal solution $\phi^*_i(x_i, s_i)$ to SMP and suppose our claim is not true. Then, there exists a triple (i, x_i, s_i) such that $\phi^*_i(x_i, s_i)$ is strictly greater than the right hand side of (15) evaluated at such an optimal solution. Since the variable $\phi_i(x_i, s_i)$ appears only in one constraint in the left hand side of (15) and this variable has a positive coefficient in the right hand side of each of the stage $i-1$ constraints (15) in which it appears, it is possible to reduce the value of this variable strictly below $\phi^*_i(x_i, s_i)$ while maintaining feasibility. However, this also reduces the claimed optimal value of the SMP objective function, since the decision variable $\phi_i(x_i, s_i)$ has a coefficient equal to 1 in this objective function. This contradicts the optimality of $\phi^*_i(x_i, s_i)$. □

As a next step, we restrict SMP by replacing the constraint set (15) with

$$\phi_i(x_i, s_i) \quad \geq \quad \max_{a \in \mathcal{A}(x_i)} r(a, s_i) + \delta \mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1}) | F_{i,i+1}\right], \forall i \in \mathcal{I}, (x_i, s_i, F_{i,i+1}) \in \mathcal{X}^D \times \prod_{j=i}^{i+1} \mathcal{F}^D_{i,j}. \tag{17}$$

That is, the constraint set (17) is obtained by expanding the first conditional expectation in (15) and listing the resulting constraints for each futures price $F_{i,i+1} \in \mathcal{F}^D_{i,i+1}$. Finally, we replace the maximization over $\mathcal{A}^D(x)$ in (17) by additional constraints for each action $a \in \mathcal{A}^D(x)$ to obtain the following Optimistic ALP (OALP; the reason for calling this ALP optimistic will become apparent soon):

$$\textbf{OALP:} \quad \min_{\phi} \sum_{i\in\mathcal{I}, x_i\in\mathcal{X}^D, s_i\in\mathcal{F}^D_{i,i}} \phi_i(x_i, s_i) \tag{18}$$

$$\text{s.t. } \phi_i(x_i, s_i) \geq \; r(a, s_i) + \delta\mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1}) | F_{i,i+1}\right],$$

$$\forall i \in \mathcal{I}, (x_i, s_i, F_{i,i+1}) \in \mathcal{X}^D \times \prod_{j=i}^{i+1} \mathcal{F}^D_{i,j}, a \in \mathcal{A}^D(x_i), \tag{19}$$

$$\phi_N(x_N, s_N) = 0, \forall x_N \in \mathcal{X}^D. \tag{20}$$

OALP is an ALP as it can be derived by using the value function approximation $\phi_i(x_i, s_i)$ from the following linear program, which is equivalent to DDP (Puterman 1994, §6.9):

$$\min_{V^D} \sum_{i\in\mathcal{I}, x_i\in\mathcal{X}^D, F_i\in\mathcal{F}^D_i} V^D_i(x_i, F_i) \tag{21}$$

$$\text{s.t. } V^D_i(x_i, F_i) \geq r(a, s_i) + \delta\mathbb{E}\left[V^D_{i+1}\left(x_i - a, \tilde{F}_{i+1}\right) | F'_i\right],$$

$$\forall i \in \mathcal{I}, (x_i, F_i) \in \mathcal{X}^D \times \mathcal{F}^D_i, a \in \mathcal{A}^D(x_i), \tag{22}$$

$$V^D_N(x_N, F_N) = 0, \;\; \forall x_N \in \mathcal{X}^D. \tag{23}$$

The decision variables of the linear program (21)-(23) are the $V^D_i(x_i, F_i)$ terms. OALP follows from replacing the variables $V^D_i(x_i, F_i)$ in (21)-(23) with the variables $\phi_i(x_i, s_i)$ and noticing that the only futures price relevant to the evolution of F'_i into s_{i+1} is $F_{i,i+1}$ (as assumed at the end of §3.3).

The analysis so far yields the following *first* key insight: SMP and hence SADP are a relaxation of OALP. That is, the first expectation in SADP has a relaxing role with respect to OALP. We now show that this relaxation has a beneficial effect. That is, although one could use an optimal OALP solution for bound computation, this is not advisable.

We start by establishing in Proposition 4.2 that OALP can be equivalently expressed as the following Optimistic ADP (OADP):

$$\textbf{OADP:} \quad \phi_i(x_i, s_i) = \max_{F_{i,i+1} \in \mathcal{F}^D_{i,i+1}} \left\{ \max_{a \in \mathcal{A}^D(x_i)} r(a, s_i) + \delta \mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1}) | F_{i,i+1}\right] \right\}, \tag{24}$$

$\forall i \in \mathcal{I}$ and $(x_i, s_i) \in \mathcal{X}^D \times \mathcal{F}^D_{i,i}$, with $\phi_N(x_N, S_N) := 0$, $\forall x_N \in \mathcal{X}^D$.

Proposition 4.2. *The optimal value function of OADP optimally solves OALP.*

Proof. OALP is feasible because the optimal value function of OADP, which exists, is a feasible solution to OALP. Further, an optimal solution to OALP must satisfy (24): Otherwise, at least one constraint of OALP would not bind, and the optimal OALP objective function value could be improved by reducing the value of an OALP decision variable without violating feasibility; the resulting feasible solution would have a lower objective function value than the assumed optimal objective function value, since all the decision variables have a positive coefficient in the OALP. This is a contradiction. □

OADP has two maximizations: The first over the set $\mathcal{F}^D_{i,i+1}$, and the second over the set $\mathcal{A}^D(x_i)$. The second maximization is analogous to the maximization in DDP. The first maximization implies that OADP treats the exogenous futures price $F_{i,i+1}$ as a choice. This is clearly unrealistic. That is, OADP relies on the *optimistic* assumption that a maximizer of the first maximization in (24), that is, a price $F_{i,i+1}$ occurs with probability one in stage i (this explains the "O" in the acronyms OADP and OALP). To emphasize the undesirable effect of this maximization we show in Proposition 4.3 that, under a mild assumption, the following continuous version of OADP has an unbounded value function in every state in stages 0 through $N-2$:

$$\phi_i(x_i, s_i) = \sup_{F_{i,i+1} \in \mathbb{R}_+} \left\{ \max_{a \in \mathcal{A}(x_i)} r(a, s_i) + \delta \mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1}) | F_{i,i+1}\right] \right\}, \tag{25}$$

$\forall i \in \mathcal{I}$ and $(x_i, s_i) \in \mathcal{X} \times \mathbb{R}_+$, with $\phi_N(x_N, s_N) = 0$, $\forall x_N \in \mathcal{X}$. This is not the case with EDP when using any reasonable forward curve evolution model, including the multi-maturity Black model (5)-(6). The mild assumption in Proposition 4.3 is that the distribution of the random variable $\tilde{s}_{i+1}$ conditional on $F_{i,i+1}$, $\tilde{s}_{i+1}|F_{i,i+1}$, is stochastically increasing in $F_{i,i+1}$ (see, e.g., Topkis 1998, Lemma 3.9.1 (b)). For example, the multi-maturity Black (1976) model (5)-(6) satisfies this property.

Proposition 4.3. *If the distribution of $\tilde{s}_{i+1}|F_{i,i+1}$ is stochastically increasing in $F_{i,i+1} \in \mathbb{R}_+$, $\forall i \in \mathcal{I}$, then the optimal value function of model (25) is unbounded in every state in stages* 0 *through* $N-2$.

Proof. Define $(\cdot)^+ := \max(0, \cdot)$. It holds that $\phi_{N-1}(x_{N-1}, s_{N-1}) = (\alpha^W s_{N-1} - c^W)^+ x$ for all $x_{N-1} \in \mathcal{X}$, and $s_{N-1} \in \mathbb{R}_+$, since $\phi_N(x_N, s_N) \equiv 0$ for all $x_N \in \mathcal{X}$. At stage $N-2$ for $x_{N-2} \in \mathcal{X} \setminus \{0\}$ we have

$$\begin{aligned}
\phi_{N-2}(x_{N-2}, s_{N-2}) &= \sup_{F_{N-2,N-1} \in \mathbb{R}_+} \left\{ \max_{a \in \mathcal{A}(x_{N-2})} r(a, s_{N-2}) + \delta \mathbb{E}\left[\phi_{N-1}(x_{N-2} - a, \tilde{s}_{N-1}) | F_{N-2,N-1}\right] \right\} \\
&= \sup_{F_{N-2,N-1} \in \mathbb{R}_+} \left\{ \max_{a \in \mathcal{A}(x_{N-2})} r(a, s_{N-2}) \right. \\
&\qquad \left. + \alpha^W \delta (x_{N-2} - a) \mathbb{E}\left[\left(\tilde{s}_{N-1} - \frac{c^W}{\alpha^W} \right)^+ | F_{N-2,N-1} \right] \right\} \\
&\geq r(0, s_{N-2}) + \alpha^W \delta x_{N-2} \sup_{F_{N-2,N-1} \in \mathbb{R}_+} \mathbb{E}\left[\left(\tilde{s}_{N-1} - \frac{c^W}{\alpha^W} \right)^+ | F_{N-2,N-1} \right] \qquad (26) \\
&= \alpha^W \delta x_{N-2} \sup_{F_{N-2,N-1} \in \mathbb{R}_+} \mathbb{E}\left[\left(\tilde{s}_{N-1} - \frac{c^W}{\alpha^W} \right)^+ | F_{N-2,N-1} \right], \qquad (27)
\end{aligned}$$

where we obtain (26) by noting that the do nothing decision, $a = 0$, is feasible in the maximization in (25), and (27) from $r(0, s_{N-2}) = 0$. The term $\mathbb{E}\left[\left(\tilde{s}_{N-1} - c^W \alpha^W \right)^+ | F_{N-2,N-1} \right]$ is an increasing function of $F_{N-2,N-1}$ under the assumption that the distribution of $\tilde{s}_{N-1}|F_{N-2,N-1}$ is stochastically increasing in $F_{N-2,N-1}$ (Topkis 1998, Corollary 3.9.1 (a)). It follows that $\phi_{N-2}(x_{N-2}, s_{N-2}) = \infty$, for all $x_{N-2} \in \mathcal{X} \setminus \{0\}$ and $s_{N-2} \in \mathbb{R}_+$. To show that $\phi_{N-2}(0, s_{N-2}) = \infty$ we follow a similar proof but use $a = C^I$ instead of the do nothing action $a = 0$.

Suppose that our claim is also true for stages $i+1$ through $N-2$. We conclude by proving our claim for stage i. Since $\phi_{i+1}(x_{i+1}, s_{i+1}) = \infty$ for all $x_{i+1} \in \mathcal{X}$ and $s_{i+1} \in \mathbb{R}_+$, it is immediate that $\delta \mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1}) | F_{i,i+1}\right] = \infty$ for all $x_i \in \mathcal{X}$, $a \in \mathcal{A}(x_i)$, and $F_{i,i+1} \in \mathbb{R}_+$. It follows that $\phi_i(x_i, s_i) = \infty$ for all $x_i \in \mathcal{X}$ and $s_i \in \mathbb{R}_+$. □

Consistent with Proposition 4.3, we have observed in computational experiments that a maximizer of the first maximization of OADP is typically the largest value in the set $\mathcal{F}^D_{i,i+1}$. These "unlikely" prices, that is, prices in the right tail of the distribution of the random variable $\tilde{F}_{i,i+1}$ conditional on $F_{0,i+1}$, determine the value function approximation used to estimate lower and upper bounds. This unrealistic value function approximation has poor bounding performance.

These observations and Proposition 4.3 suggest the following *second* key insight: When approximating DDP with OALP, the role of the relaxing expectation in SADP, that is, the first expectation in (13), is to eliminate the maximization over the prompt month futures price that is embedded in the OALP constraints for stages 0 through $N-2$. The numerical work of LMS suggests that the value function of the resulting ADP, that is SADP, has favorable bounding performance, when coupled with reoptimization for lower bound estimation.

5 The PSR Methodology

Our analysis in §4 shows that (i) SADP is a specific relaxation of OALP, and (ii) not performing such a relaxation yields value function approximations with poor bounding performance when using OALP to approximate DDP. In this section, we leverage these insights by developing our PSR methodology in §5.1. SADP is only one of the ADPs that can be obtained from our PSR approach. We apply our PSR methodology to derive novel ADPs in §§5.2-5.3; other PSR-based ADPs can be derived: Online Appendix A presents one such example. We discuss generalizations of our PSR approach in §5.4.

Our discussion in this section focuses on OALP and a version of OALP obtained from DDP by using a value function approximation analogous to the one used by OALP. However, our PSR methodology can be applied to other ALPs obtained from DDP using value function approximations that are based on different reductions of the exogenous information F_i.

5.1 Main Idea

For concreteness, we focus on OALP. Our PSR methodology includes two steps: (i) Create a partition of the OALP constraint set into the K sets $\mathcal{G}_1, \mathcal{G}_2, \ldots,$ and $\mathcal{G}_K$; (ii) replace each constraint set $\mathcal{G}_k$ by a single surrogate constraint in the sense of Glover (1968, 1975); that is, the k-th such constraint is a non-negative linear combination of the constraints in the set $\mathcal{G}_k$. More specifically, represent the constraints of set $\mathcal{G}_k$ as the system of linear inequalities $A^k z^k \geq d_k$. We choose a compatible vector of non-negative multipliers u^k, and replace $\mathcal{G}_k$ by the single constraint $u^k A^k z^k \geq u^k d_k$. Clearly, the resulting system of constraints is implied by the OALP constraints, and is thus a relaxation of OALP. Optimally solving this relaxation yields a value function approximation that can be used for bounding purposes, as discussed in §3.2.

We illustrate this approach in §§5.2-5.3. Moreover, our derivation of OALP from SADP in §4 shows that SADP can be obtained as a PSR of a math program that is equivalent to OALP. Thus, additional relaxations can be obtained by equivalently reexpressing OALP as an equivalent nonlinear math program.

5.2 A Single Price PSR and Its Equivalent ADP

In this subsection, we present a natural PSR of OALP and show that it can be formulated as an equivalent ADP. Each constraint of OALP is defined over the tuple $(i, x_i, a, s_i, F_{i,i+1})$. We partition the constraints of OALP according to the values of (i, x_i, a, s_i); that is, we have $K = |\mathcal{I}| \times |\mathcal{X}^D| \times \left(\sum_{x_i \in \mathcal{X}^D} |\mathcal{A}(x_i)|\right) \times |\mathcal{F}^D_{i,i}|$ sets in this partition, with all the constraints in each one of these K sets defined for given values of (i, x_i, a, s_i).

Our discussion following 4.3 suggests that the poor bounding performance of OADP is due to its value function approximation being determined by the largest price $F^M_{i,i+1}(s_i)$ in the set $\mathcal{F}^D_{i,i+1}(s_i)$ of all the prompt month futures prices in $\mathcal{F}^D_{i,i+1}$ given the spot price s_i: $F^M_{i,i+1}(s_i) := \max\{F_{i,i+1} : F_{i,i+1} \in \mathcal{F}^D_{i,i+1}(s_i)\}$ (if $\max\{F_{i,i+1} | F_{i,i+1} \in \mathcal{F}^D_{i,i+1}(s_i)\}$ has multiple optima, we choose as $F^M_{i,i+1}(s_i)$ any one of its maximizers). Given the pair (x_i, s_i), this suggests that an optimal OALP solution satisfies as an equality the OALP constraint corresponding to the price $F^M_{i,i+1}(s_i)$ and the optimal action associated with this price in OADP, that is, the optimal action associated with this price in the second maximization in (24).

Our first PSR is based on an intuitively likely better choice for this binding constraint. We choose this constraint to be the one corresponding to the expected prompt month futures price at time T_i given the spot price in stage i, s_i, and the maturity T_{i+1} futures price in stage 0, $F_{0,i+1}$. That is, this price is $\bar{F}_{i,i+1}(s_i) := \mathbb{E}[\tilde{F}_{i,i+1} | s_i, F_{0,i+1}]$. This price is a likely better choice than $F^M_{i,i+1}(s_i)$ as it is more "probable" than $F^M_{i,i+1}(s_i)$.

To ensure that the chosen constraint is binding at optimality, we delete from each partition set identified by (i, x_i, a, s_i) all the constraints corresponding to values of the price $F_{i,i+1}$ different from $\bar{F}_{i,i+1}(s_i)$. Therefore, the surrogate multipliers are equal to 1 when $F_{i,i+1} = \bar{F}_{i,i+1}(s_i)$ and to 0 otherwise. If $\bar{F}_{i,i+1}(s_i) \notin \mathcal{F}^D_{i,i+1}(s_i)$, then we use as a proxy the value closest to $\bar{F}_{i,i+1}(s_i)$ in

$\mathcal{F}^D_{i,i+1}(s_i)$.

Applying this PSR to OALP yields the following relaxation of its constraint set:

$$\phi_i(x_i, s_i) \geq r(a, s_i) + \delta \mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1}) | \bar{F}_{i,i+1}(s_i)\right], \forall (i, x_i, s_i) \in \mathcal{I} \times \mathcal{X}^D \times \mathcal{F}^D_{i,i}, a \in \mathcal{A}^D(x_i). \quad (28)$$

Since this constraint set is a singleton for each tuple (i, x_i, a, s_i), it is straightforward to observe that OALP with (19) relaxed by (28) is equivalent to the following ADP:

$$\textbf{ADP1:} \quad \phi_i(x_i, s_i) = \max_{a \in \mathcal{A}^D(x_i)} r(a, s_i) + \delta \mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1}) | \bar{F}_{i,i+1}(s_i)\right], \quad (29)$$

$\forall i \in \mathcal{I}$, $(x_i, s_i) \in \mathcal{X}^D \times \mathcal{F}^D_{i,i}$, with $\phi_N(x_N, s_N) := 0$, $\forall\ x_N \in \mathcal{X}^D$.

It is not hard to show that the optimal value function and an policy of ADP1 share properties analogous to the ones of EDP stated in Proposition 3.1. In particular, ADP1 has a base-stock optimal policy. This property provides theoretical support for ADP1 and allows us to compute its optimal value function more efficiently than using enumeration (see §5 in LMS).

5.3 A Two Price PSR and Its Equivalent ADP

ADP1 computes a value function approximation that in every stage depends only on the spot price, in addition to inventory. In this subsection, we discuss a richer value function approximation, which in each stage depends on the spot and prompt month futures prices, in addition to inventory. We denote $\phi_i(x_i, s_i, F_{i,i+1})$ this value function approximation in stage i.

We obtain this value function approximation from a PSR of a version of OALP with decision variables $\phi_i(x_i, s_i, F_{i,i+1})$ and constraints expressed accordingly. Our PSR of this OALP version is analogous to the one used in §5.2, with the obvious modification that $\bar{F}_{i,i+1}(s_i)$ is replaced by $\bar{F}_{i,i+2}(s_i, F_{i,i+1}) := \mathbb{E}\left[\tilde{F}_{i,i+2} | s_i, F_{i,i+1}, F_{0,i+2}\right]$. This yields the following ADP:

ADP2:

$$\begin{aligned}\phi_i(x_i, s_i, F_{i,i+1}) &= \max_{a \in \mathcal{A}^D(x_i)} r(a, s_i) + \delta \mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1}, \tilde{F}_{i+1,i+2}) | F_{i,i+1}, \bar{F}_{i,i+2}(s_i, F_{i,i+1})\right], \\ & \forall i \in \mathcal{I} \setminus \{N-2, N-1\}, (x_i, s_i, F_{i,i+1}) \in \mathcal{X}^D \times \prod_{j=i}^{i+1} \mathcal{F}^D_{i,j}, \quad (30)\end{aligned}$$

$$\begin{aligned}\phi_i(x_i, s_i, F_{i,i+1}) &= \max_{a\in\mathcal{A}^D(x_i)} r(a, s_i) + \delta\mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1})|F_{i,i+1}\right], \\ &\quad \forall i \in \{N-2, N-1\}, (x_i, s_i) \in \mathcal{X}^D \times \mathcal{F}^D_{i,i}, \qquad (31)\\ \phi_N(x_N, s_N) &:= 0,\ \forall x_N \in \mathcal{X}^D. \qquad (32)\end{aligned}$$

It is easy to show that ADP2 shares structural properties comparable to the ones of EDP stated in Proposition 3.1. As for ADP1, this provides theoretical support for ADP2 and facilitates the computation of its optimal value function.

5.4 PSR Generalizations

Generalizations of our PSR methodology are possible. Consider OALP. Although the first step in our approach is restricted to considering *partitions* of the constraint set of OALP, our relaxation procedure easily extends to the case when the sets $\mathcal{G}_1, \mathcal{G}_2, \ldots$, and $\mathcal{G}_K$ do not form such a partition. That is, we could consider surrogate relaxations rather than *Partitioned* surrogate relaxations. However, for a general choice of these sets, the resulting relaxed linear/math program may not be representable as an ADP, that is, a model analogous to ADP1, ADP2, or SADP. Proposition 5.1 provides sufficient conditions for the choice of these sets to yield such an ADP. For ease of exposition, we state our conditions with reference to OALP, but extensions to ALPs with approximate value functions based on different reductions of the forward curve are straightforward. We omit the proof of Proposition 5.1, as it is similar to the proofs of Propositions 3.1 and 4.2. Proposition 5.1 holds for ADP1 and ADP2 (with OALP modified as stated earlier for ADP2).

Proposition 5.1. *If each constraint in each set $\mathcal{G}_k$, $k \in \{1, \ldots, K\}$, shares the same triple (i, x_i, s_i), then the linear program resulting from the PSR of OALP based on the sets $\mathcal{G}_k$, $k \in \{1, \ldots, K\}$, has an equivalent ADP representation. Further, the resulting ADP shares analogous to the ones stated in Proposition 3.1.*

6 Structural Analysis of the ADP1 and ADP2 Optimal Value Functions and their Associated Bounds

In this section, we investigate how the optimal value functions of ADP1 and ADP2 relate to the optimal value function of EDP, and the likely quality of their resulting greedy lower and dual upper bounds. For simplicity, we consider versions of ADP1 and ADP2 with continuous price sets. With a slight abuse of notation, we continue to refer to these ADPs as ADP1 and ADP2.

In the general case, it is easy to show that ADP1 and ADP2 coincide with EDP for problems with up to two stages ($N = 2$) and three stages ($N = 3$), respectively. This may not be true for an arbitrary number of stages. We thus analyze the easier special case, studied by Secomandi (2011), in which the storage asset is *fast* (that is, $-C^I = C^W = \bar{x}$) and there are no *frictions* (that is, $\alpha^W = \alpha^I = 1$ and $c^W = c^I = 0$). In this case, Secomandi (2011) shows that EDP is tractable, since its exact value function can be written as $V_i(x_i, F_i) = \gamma_i(F_i)\bar{x} + s_i x_i$, where $\gamma_i(F_i) := (\delta F_{i,i+1} - s_i)^+ + \sum_{j=i+1}^{N-2} \delta^{j-i} \mathbb{E}\left[(\delta \tilde{F}_{j,j+1} - \tilde{s}_j)^+ | F_i'\right]$. That is, this function is linear in inventory with intercept $\gamma_i(F_i)\bar{x}$ and slope s_i. An optimal policy thus simply involves a comparison of the spot price and the discounted prompt month futures price in every stage and state (Secomandi 2011).

Although heuristics are not needed when the storage asset is fast and frictionless, it is insightful to investigate ADP1 and ADP2 in this restricted case. Proposition 6.1 characterizes the optimal value functions of ADP1 and ADP2 in this case. We omit the proof of Proposition 6.1 as it follows from a straightforward induction argument. We define the functions $\gamma_i^{\phi}(s_i)$ and $\gamma_i^{\phi}(s_i, F_{i,i+1})$ as follows:

$$\gamma_i^{\phi}(s_i) := \left(\delta \bar{F}_{i,i+1}(s_i) - s_i\right)^+ + \delta \mathbb{E}\left[\gamma_{i+1}^{\phi}(\tilde{s}_{i+1}) | \bar{F}_{i,i+1}(s_i)\right],$$

$$\gamma_i^{\phi}(s_i, F_{i,i+1}) := (\delta F_{i,i+1} - s_i)^+ + \delta \mathbb{E}\left[\gamma_{i+1}^{\phi}(\tilde{s}_{i+1}, \tilde{F}_{i+1,i+2}) | F_{i,i+1}, \bar{F}_{i,i+2}(s_i, F_{i,i+1})\right].$$

Notice that in general the functions $\gamma_i^{\phi}(s_i)$ and $\gamma_i^{\phi}(s_i, F_{i,i+1})$ are not equal to the function $\gamma_i(F_i)$.

Proposition 6.1. *When the storage asset is fast and there are no frictions, the ADP1 optimal value function is $\phi_i(x_i, s_i) = \gamma_i^{\phi}(s_i)\bar{x} + s_i x_i$, $\forall i \in \mathcal{I}$ and $(x_i, s_i) \in \mathcal{X} \times \mathbb{R}_+$, and the ADP2 optimal value function is $\phi_i(x_i, s_i, F_{i,i+1}) = \gamma_i^{\phi}(s_i, F_{i,i+1})\bar{x} + s_i x_i$, $\forall i \in \mathcal{I}$ and $(x_i, s_i, F_{i,i+1}) \in \mathcal{X} \times \mathbb{R}_+^2$.*

Proposition 6.1 shows that the value function slopes of ADP1, ADP2, and EDP are all equal for a fast and frictionless storage asset. This implies that in this case using the ADP1 and ADP2 optimal value functions in (8) yields an optimal action. Hence, the corresponding greedy lower bounds estimated by Monte Carlo simulation are tight. Interestingly, the policy obtained from solving ADP2, rather than using (8), is also optimal. In contrast, this is not true for ADP1. This is because the slope of the ADP1 continuation value function, that is, $\delta\mathbb{E}\left[\phi_{i+1}(\cdot,\tilde{s}_{i+1})|\bar{F}_{i,i+1}(s_i)\right]$ is $\delta\mathbb{E}\left[\tilde{s}_{i+1}|\bar{F}_{i,i+1}(s_i)\right] = \delta\bar{F}_{i,i+1}(s_i)$, whereas the one used both by ADP2 and EDP is $\delta\mathbb{E}\left[\tilde{s}_{i+1}|F_{i,i+1}\right] = \delta F_{i,i+1}$.

The intercept of the ADP1 and ADP2 optimal value functions do not play a role in determining an action in (8). Thus, such an intercept does not affect the estimation of a greedy lower bound. This is also true for the estimation of a dual upper bound, as now explained. For a fast and frictionless storage asset, Proposition 6.1 implies that the exact dual penalty (9) is

$$\begin{aligned}
p_i(x_i,a,F_{i+1},F_i') &= V_{i+1}(x_i-a,F_{i+1}) - \mathbb{E}\left[V_{i+1}(x_i-a,\tilde{F}_{i+1})|F_i'\right] \\
&= (s_{i+1}-F_{i,i+1})(x_i-a) \\
&\quad +\bar{x}\left\{(\delta F_{i+1,i+2}-s_{i+1})^+ - \mathbb{E}\left[(\delta\tilde{F}_{i+1,i+2}-\tilde{s}_{i+1})^+|F_i'\right]\right\} \\
&\quad +\bar{x}\left\{\sum_{j=i+2}^{N-2}\delta^{j-i-1}\mathbb{E}\left[(\delta\tilde{F}_{j,j+1}-\tilde{s}_j)^+|F_{i+1}'\right]\right. \\
&\qquad \left. -\sum_{j=i+2}^{N-2}\delta^{j-i-1}\mathbb{E}\left[(\delta\tilde{F}_{j,j+1}-\tilde{s}_j)^+|F_i'\right]\right\}.
\end{aligned} \tag{33}$$

The analogous dual penalty derived from using the ADP1 optimal value function is

$$\begin{aligned}
p_i^{\phi}(x_i,a,s_{i+1},F_{i,i+1}) &= \phi_{i+1}(x_i-a,s_{i+1}) - \mathbb{E}\left[\phi_{i+1}(x_i-a,\tilde{s}_{i+1})|F_i'\right] \\
&= (s_{i+1}-F_{i,i+1})(x_i-a) \\
&\quad +\bar{x}\left\{(\delta\bar{F}_{i+1,i+2}(s_{i+1})-s_{i+1})^+ - \mathbb{E}\left[(\delta\bar{F}_{i+1,i+2}(\tilde{s}_{i+1})-\tilde{s}_{i+1})^+|F_i'\right]\right\} \\
&\quad +\bar{x}\left\{\delta\mathbb{E}\left[\gamma_{i+2}^{\phi}(\tilde{s}_{i+2})|\bar{F}_{i+1,i+2}(s_{i+1})\right]\right. \\
&\qquad \left. -\mathbb{E}\left[\delta\mathbb{E}\left[\gamma_{i+2}^{\phi}(\tilde{s}_{i+2})|\bar{F}_{i+1,i+2}(\tilde{s}_{i+1})\right]|F_i'\right]\right\}.
\end{aligned} \tag{34}$$

Comparing (33) and (34) reveals that, in general, they agree only with respect to the slope related term $(s_{i+1}-F_{i,i+1})(x_i-a)$. A similar statement holds when the dual penalty is specified using the ADP2 optimal value function. However, the dual upper bounds estimated using the optimal value functions of ADP1 and ADP2 are tight for the fast and frictionless case, because, conditional on F_0', the expectation of the terms that depend on $\bar{x}$ in (34) is zero.

Although this analysis is specific to the case of no frictions, it has broader implications. For a fast storage asset, the greedy lower bounds and dual upper bounds estimated using the ADP1 and ADP2 optimal value functions are likely to be close to the EDP optimal value function in the initial stage and state when the frictions are small, which is the case for the crude oil instances that we consider in §8.4 (small frictions are typical in practice).

7 Computational Complexity

In this section, we discuss the computational complexity of obtaining the ADP1 and ADP2 optimal value functions, and estimating their corresponding greedy lower and dual upper bounds. This complexity depends on the specific technique used for discretizing the relevant price sets. Our computational study in §8 assumes that EDP is formulated using the multi-maturity Black (1976) price model (5)-(6). We thus discretize this model via Rubinstein (1994) binomial lattices, and focus our analysis on this discretization approach. However, other discretization methods may be used, e.g., some of those discussed by Levy (2004, Chapter 12).

Consider ADP1. We obtain the set $\mathcal{F}^D_{i,i}$, that is, we discretize $\mathbb{R}_+$, by evolving the time 0 futures price $F_{0,i}$ using a two-dimensional Rubinstein binomial tree. Let m_i be the number of time steps used to discretize the time interval $[0, T_i]$. Building this lattice results in a set $\mathcal{F}^D_{i,i}$ with $m_i + 1$ values. This requires $O(m_i)$ operations.

We proceed to analyze the complexity of computing the ADP1 optimal value function. At each stage i, this entails executing the following steps:

Step 1: Determine a probability mass function with support on $\mathcal{F}^D_{i+1,i+1}$ for the random variable $\tilde{s}_{i+1}|\bar{F}_{i,i+1}(s_i)$ for each $s_i \in \mathcal{F}^D_{i,i}$;

Step 2: Compute the optimal ADP1 basestock targets for each $s_i \in \mathcal{F}^D_{i,i}$;

Step 3: Evaluate $\phi_i(x_i, s_i)$ for all the states $(x_i, s_i) \in \mathcal{X}^D \times \mathcal{F}^D_{i,i}$.

In step 1, we evolve a two-dimensional Rubinstein lattice, starting from each $\bar{F}_{i,i+1}(s_i)$ referred to as the transition lattice, by using m time steps to discretize the interval $[T_i, T_{i+1}]$. Each $\bar{F}_{i,i+1}(s_i)$ can be computed in closed-form in $O(1)$ operations under the price model (5)-(6). Each transition lattice yields a discretization of s_{i+1} with $m + 1$ values. Building all the m_i transition lattices thus takes $O(m_i \cdot m)$ operations. To obtain the distribution of $\tilde{s}_{i+1}|\bar{F}_{i,i+1}(s_i)$ with support on $\mathcal{F}^D_{i+1,i+1}$, we project each price s_{i+1} in each transition lattice onto the set $\mathcal{F}^D_{i+1,i+1}$ by rounding each price

s_{i+1} to the closest spot price in $\mathcal{F}^D_{i+1,i+1}$. Since the s_{i+1} values in each transition lattice and the set $\mathcal{F}^D_{i+1,i+1}$ are sorted, this projection can be done in a total of $O(m_{i+1} \cdot m)$ operations at stage i. Therefore, the time complexity for step 1 at stage i is $O(m_i \cdot m + m_{i+1} \cdot m)$.

Executing step 2 requires performing the maximization in (29) at inventory levels 0 and $\bar{x}$ with injection and withdrawal capacities relaxed. This requires $O(m_i \cdot |\mathcal{X}^D| \cdot m)$ operations. Executing step 3 also requires $O(m_i \cdot |\mathcal{X}^D| \cdot m)$ operations. Therefore, computing $\phi_i(x_i, s_i)$ for all the states $(x_i, s_i) \in \mathcal{X}^D \times \mathcal{F}^D_{i,i}$ in stage i requires $O(m \cdot (m_i + m_{i+1} + 2 \cdot m_i \cdot |\mathcal{X}^D|))$ operations. Using $m' := \max_{i \in \mathcal{I}} m_i$, this simplifies to $O(m' \cdot |\mathcal{X}^D| \cdot m)$ operations, since $|\mathcal{X}^D| \geq 2$.

Thus, for an N-stage problem, computing the ADP1 optimal value function requires $O(N \cdot m' \cdot |\mathcal{X}^D| \cdot m)$ operations.

Let n_s denote the number of price sample paths used in a Monte Carlo simulation for estimating a greedy lower bound and dual upper bound. Given the ADP1 optimal value function, a simple analysis shows that estimating these bounds requires $O(n_s \cdot N \cdot \log m' + n_s \cdot N \cdot |\mathcal{X}^D| \cdot m)$ and $O(n_s \cdot N \cdot |\mathcal{X}^D| \cdot \log m' + n_s \cdot N \cdot |\mathcal{X}^D|^2 \cdot m)$ operations ($O(\log m')$ operations are needed by binary search, which we use when projecting a transition lattice).

For ADP2, we determine the set $\mathcal{F}^D_{i,i} \times \mathcal{F}^D_{i,i+1}$ for each stage i using a three dimensional Rubinstein lattice. We also use three dimensional binomial lattices and projections to obtain the joint probability mass function of each random pair $(\tilde{s}_{i+1}, \tilde{F}_{i+1,i+2})$ conditional on the pair $(F_{i,i+1}, \bar{F}_{i,i+2}(s_i, F_{i,i+1}))$ on the support $\mathcal{F}^D_{i+1,i+1} \times \mathcal{F}^D_{i+1,i+2}$. An analysis similar to the one for ADP1 shows that we can compute the ADP2 optimal value function in $O(N \cdot m'^2 \cdot |\mathcal{X}^D|^2 \cdot m^2)$ operations and estimate a greedy lower bound and a dual upper bound in $O(n_s \cdot N \cdot \log m' \cdot m + n_s \cdot N \cdot |\mathcal{X}^D| \cdot m^2)$ and $O(n_s \cdot N \cdot |\mathcal{X}^D| \cdot \log m' \cdot m + n_s \cdot N \cdot |\mathcal{X}^D|^2 \cdot m^2)$ operations, respectively.

The top part of Table 1 summarizes the results of our computational complexity analysis for ADP1 and ADP2. Estimating dual upper bounds is more costly than estimating greedy lower bounds. This is due to the computation of the dual value function in (10) at each inventory level in the set $\mathcal{X}^D$ and for all the stages in set $\mathcal{I}$ given a price sample path P_0. Typical values of the parameters n^s, $|\mathcal{X}^D|$, and m' satisfy $n^s \cdot |\mathcal{X}^D| \geq m'$. Hence, estimating dual upper bounds is also more costly operation than computing the optimal value functions of ADP1 and ADP2; for example, this is the case in our computational experiments discussed in §8.

It is important to emphasize that the computational complexity results of solving our ADPs

Table 1: Operations count of estimating a greedy lower bound and a dual upper bound with ADP1, ADP2 and DDP.

	ADP1	ADP2
Value Function	$O(N \cdot m' \cdot \lvert\mathcal{X}^D\rvert \cdot m)$	$O(N \cdot (m')^2 \cdot \lvert\mathcal{X}^D\rvert \cdot m^2)$
Lower Bound	$O\left(n_s \cdot N \cdot \left[\log m' + \lvert\mathcal{X}^D\rvert \cdot m\right]\right)$	$O\left(n_s \cdot N \cdot m \cdot \left[\log m' + \lvert\mathcal{X}^D\rvert \cdot m\right]\right)$
Upper Bound	$O\left(n_s \cdot N \cdot \lvert\mathcal{X}^D\rvert \cdot \left[\log m' + \lvert\mathcal{X}^D\rvert \cdot m\right]\right)$	$O\left(n_s \cdot N \cdot \lvert\mathcal{X}^D\rvert \cdot m \cdot \left[\log m' + \lvert\mathcal{X}^D\rvert \cdot m\right]\right)$
	DDP	
Value Function	$O(N \cdot (m')^N \cdot \lvert\mathcal{X}^D\rvert \cdot m^N)$	
Lower Bound	$O\left(n_s \cdot m^N \cdot \left[\log m' + \lvert\mathcal{X}^D\rvert \cdot m\right]\right)$	
Upper Bound	$O\left(n_s \cdot \lvert\mathcal{X}^D\rvert \cdot m^N \cdot \left[\log m' + \lvert\mathcal{X}^D\rvert \cdot m\right]\right)$	

and estimating bounds using their optimal value functions is linear in N, even if we use the multi-maturity price model (5)-(6), which is equivalent to an N factor model. In contrast, solving DDP exactly, also using Rubinstein lattices to discretize the price model (5)-(6), and employing the resulting DDP optimal value function for bound computation requires an exponential number of operations in N, as summarized in the bottom part of Table 1.

Moreover, we are not aware of any methods to solve EDP exactly in the general case, even when using a one factor price model rather than the price model (5)-(6) which, as stated above, is equivalent to an N factor model.

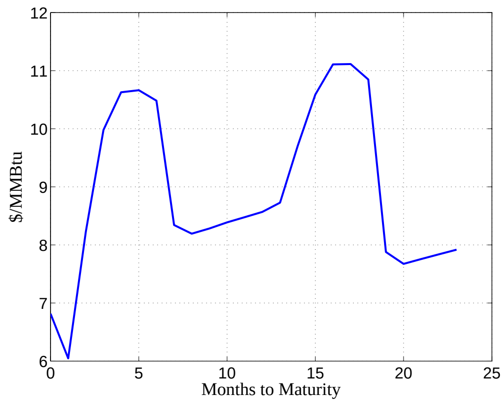
8 Computational Results

In this section, we present our computational results. We discuss our benchmarking instances in §8.1. We specify the values used for our state space discretization in §8.2. We investigate the performance of our PSR methodology in §§8.3-8.4. Online Appendix C reports tables that display all the results discussed in §§8.3-8.4.

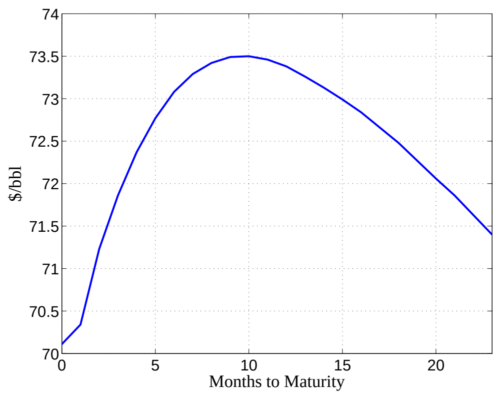
Our experiments are based on the following computational setup: A 64 bits PowerEdge R515 with twelve AMD Opteron 4176 2.4GHz processors, of which we used only one, with 64GB of memory, the Linux Fedora 15 operating system, and the g++ 4.6.1 20110908 (Red Hat 4.6.1-9) compiler. The SADP results that we report are obtained using the code of LMS run under our computational setup.

8.1 Instances

We consider natural gas and crude oil instances. These instances are representative of two common types of commodity storage encountered in practice. They differ in two aspects. Their first difference is the type of commodity forward curve. Natural gas consumption exhibits seasonal patterns, which are reflected in a forward curve with higher prices in the winter months than in the summer months. In contrast, seasonality in crude oil forward curves is less apparent. Figure 1 illustrates these forward curves on two sample dates – natural gas and crude oil futures prices are expressed in dollars per million British thermal units ($/MMBtu) and dollars per barrel ($/bbl), respectively. The second difference is related to the operational characteristics of the storage facility. Storage facilities for natural gas can be *slow*, that is, they can have constrained injection and withdrawal capacities, $-C^I < \bar{x}$ and/or $C^W < \bar{x}$, even with a monthly review period (the length of time between two adjacent stages in our ADPs). In contrast, crude oil storage facilities are *fast*, that is, $-C^I = C^W = \bar{x}$, even with a daily review period (Griffin 2011).



(a) Natural gas (June 1, 2006)



(b) Crude oil (August 31, 2006)

Figure 1: The natural gas and crude oil forward curves (closing prices on June 1, 2006 and August 31, 2006, respectively).

For natural gas, we use the twelve 24-stage instances of LMS, created using data from the New York Mercantile Exchange (NYMEX) and the energy trading literature. These instances have a monthly review period. Each instance is identified by a season, one of Spring, Summer, Fall, and Winter, and one of three injection and withdrawal capacity pairs, with their labels 1, 2, and 3 denoting a heavy, intermediate, and mild capacity restriction, respectively. These instances are based on the multi-maturity Black model (5)-(6).

For crude oil, we create twelve 24-stage instances using historical prices of West Texas Intermediate crude oil futures, which are traded on NYMEX. As for the natural gas instances, each stage corresponds to a monthly review period, which is consistent with our focus on the value of a storage asset due to monthly volatility. These instances are defined using the closing forward curve on the first trading date of each month in 2006. We also use the multi-maturity Black model (5)-(6) on these instances. We estimate its volatilities and correlation matrix from historical futures prices observed between 2002 and 2006. These forward curves and the estimated volatilities and correlation matrix are available in Online Appendix B. We set both the injection and withdrawal marginal costs, c^I and c^W, to 0.001\$/bbl, and the injection and withdrawal losses, α^I and α^W, to zero (Griffin 2011).

We use a risk free interest rate equal to 4.74%, 5.05%, 5.01%, and 4.87% for the Spring, Summer, Fall, and Winter natural gas instances, respectively, as in LMS, and 4.74% for the crude oil instances, that is, the minimum of the natural gas risk free interest rates.

8.2 State Space Discretization

Recall that ADP1 and ADP2 are formulated on discretized state and action spaces: $\mathcal{X}^D \times \mathcal{F}^D_{i,i}$ and $\mathcal{A}^D(x)$ for ADP1, and $\mathcal{X}^D \times \mathcal{F}^D_{i,i} \times \mathcal{F}^D_{i,i+1}$ and $\mathcal{A}^D(x)$ for ADP2. Based on the discussion preceding Proposition 5.1 and this proposition, we obtain $\mathcal{X}^D$ by discretizing the $[0, 1 =: \bar{x}]$ inventory set $\mathcal{X}$ into 21 equally spaced points for the natural gas instances, as in LMS, and 2 equally spaced points (0 and 1) for our crude oil instances. Computing the discretized price sets requires choosing values for the parameters m_i and m (see §7). On the natural gas instances, as in LMS, we set m_i equal to 500 and m equal to 20 when solving ADP1, as this facilitates comparing our bounds with those of LMS. We use the same values for ADP2, but also apply lattice restrictions (Levy 2004) to deal with the computational burden of discretizing $\mathbb{R}^2_+$. Specifically, we restrict each value of the spot price $s_i \in \mathcal{F}^D_{i,i}$ to be less than or equal to 4 times the time zero futures price with maturity T_i, $F_{0,i}$. This implies a restriction for the values of the prompt month futures prices in each set $\mathcal{F}^D_{i,i+1}$. This approach, standard in computational finance, is effective in our application: Our estimated lower and upper bounds change by less than 0.2% when we add this restriction, and we obtain an order of magnitude speed up. When computing reoptimized lower bounds (see §3.2) we use a coarser discretization by setting $m_i = m = 5$, as in LMS. On the crude oil instances, we also set $m = 20$ but we consider a range of values for m_i, because we observed that the choice of this parameter value affected the quality of the estimated bounds, in particular the dual upper bound. Reoptimization is performed under the same setting used for natural gas.

8.3 Results on the Natural Gas Instances

We now compare the bounding performance of ADP1, ADP2, and SADP on the natural gas instances. As in LMS, we use $10,000$ sample paths to obtain all the reported greedy lower and dual upper bound estimates in state $(0, F_0)$ in stage 0. The numerical results in LMS suggest that the dual upper bound estimated using the SADP optimal value function is almost tight on the Spring, Summer, and Fall instances, but looser on the Winter instances. Thus, improvements on these upper bounds may be possible only on the Winter instances. The SADP based lower bound estimates obtained without reoptimization are fairly loose on all the considered instances; their reoptimized versions become almost tight on the Spring, Summer, and Fall instances, but not on the Winter instances, where there is an optimality gap of up to 6%. Therefore, it may be possible to improve on the SADP based lower bound estimates without reoptimization on all the instances and their reoptimized versions on the Winter instances.

Dual Upper Bounds. We denote by UBS, UB1, and UB2 the dual upper bound estimates associated with SADP, ADP1, and ADP2, respectively. Figure 2 reports UBS and UB1 as percentages of UB2, as UB2 is tighter than both UBS and UB1. The error bars in this figure indicate the standard errors of UBS and UB1 normalized by UB2. UBS and UB1 match on all the instances after accounting for sampling variability (the UBS and UB1 standard errors vary between 0.36% and 0.72% and 0.36% and 0.87% of UB2, respectively; the UB2 standard errors range from 0.30% to 0.65% of UB2). Remarkably, UB2 is smaller than both UBS and UB1 by an average of 2.82% on the Winter instances, while its improvement is more contained on the other instances (as expected).

Greedy Lower Bounds. We denote by LBS, LB1, and LB2 the lower bound estimates obtained using SADP, ADP1, and ADP2, respectively (LB1 and LB2 are greedy lower bound estimates; as in LMS, LBS is the estimated value of the SADP optimal policy). Figure 3 shows these estimates normalized by UB2. The error bars in this figure indicate the standard errors of these estimates relative to UB2. LBS and LB1 are within standard error of each other on the Spring, Summer, and Fall instances, while LB1 is weaker than LBS by no more than 2.44% of UB2 on the Winter instances

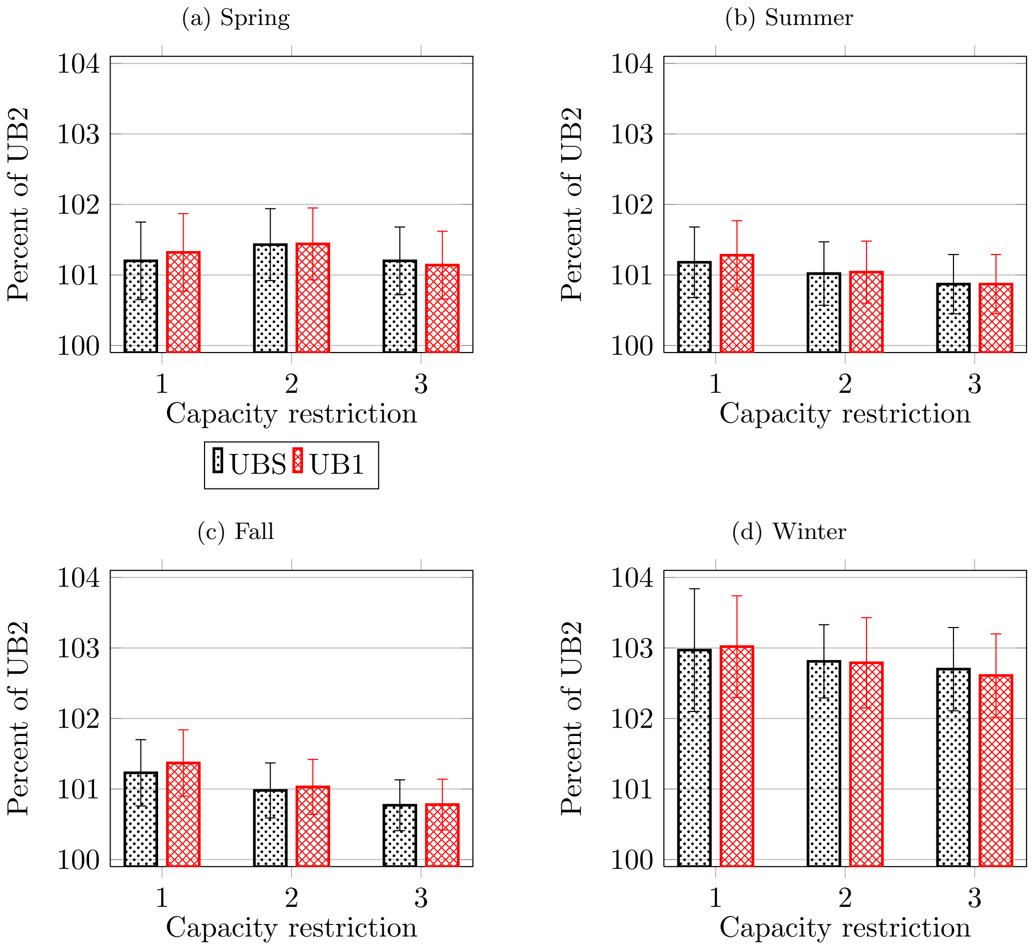


Figure 2: The estimated upper bounds on the natural gas instances.

(the LBS and LB1 standard error ranges are 0.93-1.68% and 0.97-1.43% of UB2, respectively). Interestingly, LB2 outperforms both LBS and LB1 on all the instances: LB2 improves on LBS by 2-3.36% across the Spring, Summer, and Fall instances, and by 6.72-8.43% on the Winter instances. The improvements of LB2 on LB1 are similar on the Spring, Summer, and Fall instances, but are larger on the Winter instances. The standard errors of LB2 vary between 0.92% and 1.62% of UB2. These results suggest that ADP2 is a fundamentally better model than both SADP and ADP1, with maximum suboptimality gaps of 3.03% of UB2 on the Spring, Summer, and Fall instances, and 9.03% of UB2 on the Winter instances. In contrast, these suboptimalities are 5.77% and 17.46% for SADP, and 6.11% and 19.89% for ADP1.

We denote by RLBS, RLB1, and RLB2 the estimates of the reoptimized versions of LB2, LB1, and LB2, respectively. Figure 4 displays these reoptimized lower bound estimates relative to UB2.

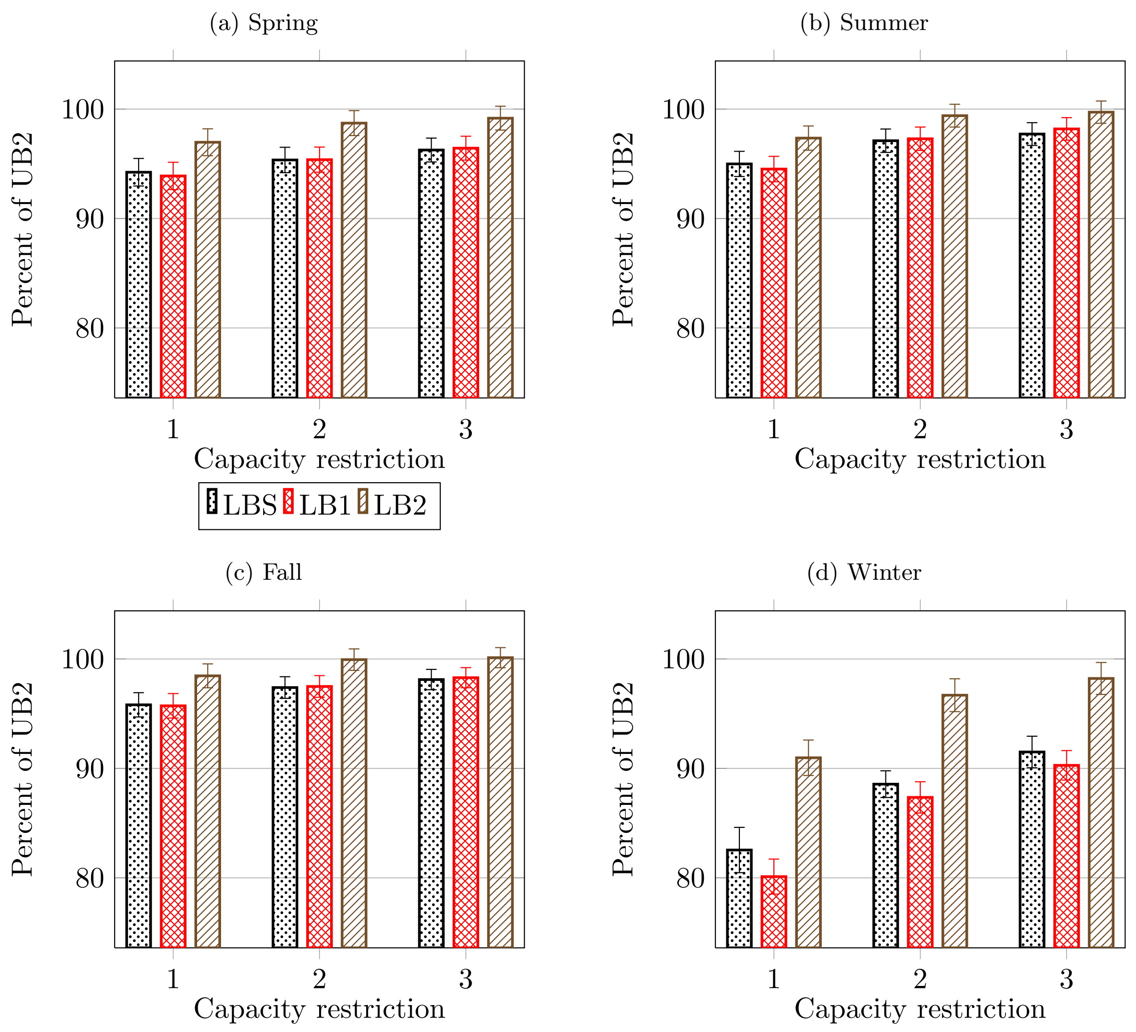


Figure 3: The estimated lower bounds on the natural gas instances (no reoptimization).

The standard errors of RLBS, RLB1, and RLB2 are spread between 0.94% and 1.73% of UB2. RLBS, RLB1, and RLB2 are almost tight on the Spring, Summer, and Fall instances, which implies that their respective greedy policies are essentially optimal on these instances (different from the policy associated with LBS, the one corresponding to RLBS is of the greedy type). RLB2 is slightly better than RLBS and RLB1 on the Winter instances, with a maximum suboptimality gap of 2.38% of UB2 compared to 3.51% for RLBS and 2.58% for RLB1. Further, LB2 is worse than RLB2 by a maximum of 6.65% of UB2 on all the instances, while LBS is below RLBS by 2.29-13.94% and LBS falls short of RLB1 by 2.07 and 17.31% on all the instances. These figures suggest that reoptimization is less critical for ADP2 than it is both for SADP and ADP1.

CPU Times. The run times required to solve ADP1 range from 0.11 to 0.12 CPU seconds,

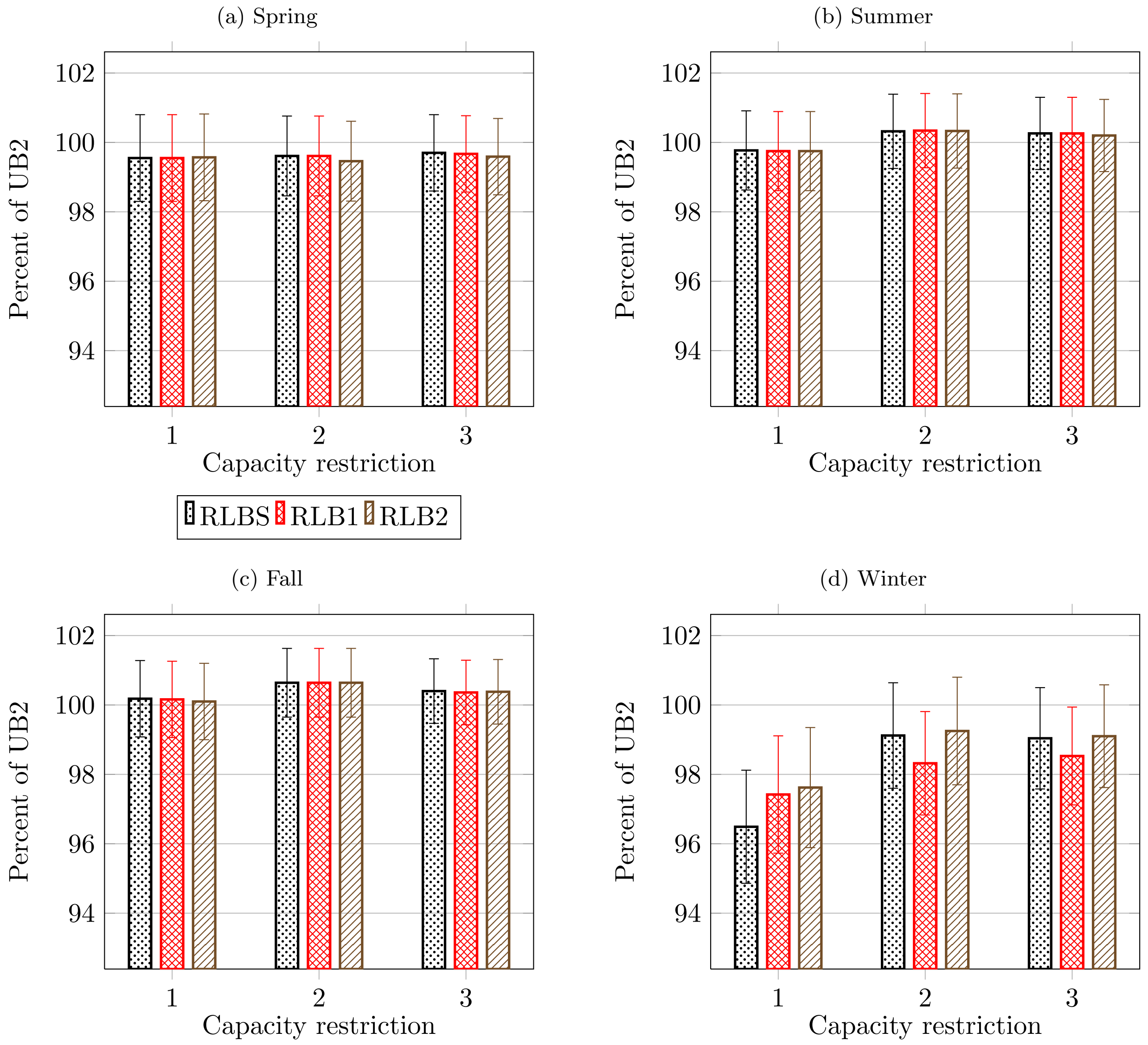


Figure 4: The estimated reoptimized lower bounds on the natural gas instances.

while the ones for SADP are between 120.16 and 121.85 CPU seconds. Thus, solving ADP1 is at least 1,000 times faster than solving SADP on all the instances. The ADP1 overall run times, that is, also including the CPU times for bound estimation, are spread between 10.21 and 17.16 CPU seconds. The SADP overall run times are between 271.83 and 313.59 CPU seconds. Therefore, the overall ADP1 CPU run time is at least an order of magnitude smaller than the one of SADP on all the instances.

The CPU times needed to solve ADP2 vary from 36.20 to 52.56 CPU seconds. The ADP2 overall run times (with restricted lattices) range from 153.73 to 225.2 CPU seconds, which is 23.90% to

47.5% less than the SADP CPU times. Thus, using ADP2 we are able to compute improved bounds in a faster fashion compared to SADP. However, solving ADP2 is 12 to 16 times slower than solving ADP1.

Computing RLBS takes between 543.5 and 619.23 CPU seconds, while this range for RLB1 is 89.75-92.60 CPU seconds, that is, roughly 6 times shorter. This run time range for RLB2 is 1,222.40-1,247.53 CPU seconds, which is roughly twice and one order of magnitude larger than the RLBS and RLB1 run time ranges.

8.4 Results on the Crude Oil Instances

We investigate the bounding performance when using ADP1 on the crude oil instances. We do not consider ADP2 due to the near optimal bounding performance when using ADP1 (we have however verified using ADP2 also yields near optimal bounds, but the ADP2 computational requirement is higher than the one of ADP1). As in the natural gas instances, we use $10,000$ sample paths to estimate our greedy lower and dual upper bounds on the value of storage in state $(0, F_0)$ in stage 0. Given that our crude oil instances involve a fast storage asset with small frictions, from our discussion in §6, we expect almost tight greedy lower bounds and dual upper bounds estimates. Without reoptimization, we consider the following values for m_i: 500, 1,000, 10,000, and 20,000.

Dual Upper Bounds. Across instances, the range of the percentage ratios of UB1 for m_i equal to 500, 1,000, and 10,000, respectively, and UB1 for m_i equal to $20,000$ is 108.81-114.53%, 104.50-107.79%, and 100.20-100.34% (the range of the UB1 standard error for m_i equal to 500, 1,000, 10,000, and 20,000 as a percentage of UB1 for m_i equal to $20,000$ is 0.31-0.44%, 0.30-0.42%, 0.30-0.40%, and 0.30-0.40%, respectively). This shows that the choice of the value of m_i is important to improve the quality of UB1 on these instances.

Greedy Lower Bounds. LB1 normalized by UB1 for m_i equal to $20,000$ is at least 97.78%, 97.81%, 98.92%, and 99.15% for m_i equal to 500, 1,000, 10,000, and 20,000, respectively (the range of these LB1 standard error as a percentage of this UB1 is 1.12-1.69%, 1.13-1.68%, 1.12-1.68%, and 1.12-1.68%). Thus, increasing the value of m_i from 500 to 20,000 slightly improves LB1 on these instances. RLB1 normalized by UB1 for m_i equal to $20,000$ is at least 99.28% across all our crude oil instances, with the RLB1 standard error spread between 1.12% and 1.68% of this UB1. Although LB1 for the considered m_i values is almost tight, reoptimization can help.

CPU Times. Across the considered m_i values, solving ADP1 requires between 0.01 and 0.14 CPU seconds, and the overall ADP1 run time is spread between 6.10 and 7.37 CPU seconds. Computing RLB1 requires about 28 CPU seconds.

9 Conclusions

Real option management of commodity storage assets is an important practical problem that, in general, gives rise to an intractable MDP when using high dimensional models of commodity forward curve evolution. We propose a novel approximate dynamic programming methodology, based on partitioned surrogate relaxations of approximate linear programs, to obtain value function approximations to this MDP. Application of our approach yields two new ADPs, as well as the known SADP as a special case. We test our ADPs on existing natural gas and new crude oil storage instances.

Our focus in this paper is on commodity storage. However, our PSR methodology is potentially relevant for the approximate solution of other real and financial option valuation and management problems (see the discussion in §1). It would be interesting to apply our methodology to such problems, also using different types of forward curve evolution models and/or value function approximations. Moreover, in this paper we apply our PSR approach using exogenously chosen partitions and multipliers. An interesting direction for additional research would be optimizing (in some sense) the partitions and surrogate multipliers of our PSR methodology. This may improve the quality of the greedy lower and dual upper bounds obtained with the value function approximations of the resulting ALPs/ADPs.

Acknowledgments

The authors thank Alan Scheller-Wolf for many helpful comments. This research was supported by NSF grant CMMI1129163.

References

D. Adelman. A price-directed approach to stochastic inventory/routing. *Operations Research*, 52(4):499–514, 2004.

D. Adelman. Math programming approaches to approximate dynamic programming. Tutorial, INFORMS Annual Meeting, Pittsburgh, PA, USA, 2006.

D. Adelman. Dynamic bid prices in revenue management. *Operations Research*, 55(4):647–661, 2007.

D. Adelman and D. Klabjan. Computing near optimal policies in generalized joint replenishment. *INFORMS Journal on Computing*, Forthcoming, 2011.

L. Andersen and M. Broadie. Primal-dual simulation algorithm for pricing multidimensional American options. *Management Science*, 50(9):1222–1234, 2004.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, Belmont, MA, USA, 1996.

J. Birge. Quasi-convex stochastic dynamic programming. Working paper, Chicago Univ., 2011.

F. Black. The pricing of commodity contracts. *Journal of Financial Economics*, 3(1-2):167–179, 1976.

A. Boogert and C. De Jong. Gas storage valuation using a Monte Carlo method. *The Journal of Derivatives*, 15(3):81–98, 2008.

A. Boogert and C. De Jong. Gas storage valuation using a multifactor price process. *The Journal of Energy Markets*, 4(4), 2011.

D. B. Brown, J. E. Smith, and P. Sun. Information relaxations and duality in stochastic dynamic programs. *Operations Research*, 58(4):1–17, 2010.

R. Carmona and M. Ludkovski. Valuation of energy storage: An optimal switching approach. *Quantitative Finance*, 10(4):359–374, 2010.

H.S. Chang, M.C. Fu, J. Hu, and S.I. Marcus. *Simulation-based Algorithms for Markov Decision Processes.* Springer-Verlag, London, UK, 2007.

Z. Chen and P.A. Forsyth. A semi-Lagrangian approach for natural gas storage valuation and optimal operation. *SIAM Journal on Scientific Computing*, 30(1):339–368, 2007.

G. Cortazar, M. Gravet, and J. Urzua. The valuation of multidimensional American real options using the LSM simulation method. *Computers & Operations Research*, 35(1):113–129, 2008.

D. P. de Farias and B. Van Roy. On constraint sampling for the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29(3):462–478, 2001.

D. P. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

D. P. de Farias and B. Van Roy. A cost-shaping linear program for average-cost approximate dynamic programming with performance guarantees. *Mathematics of Operations Research*, 31(3):597–620, 2006.

V. V. Desai, V. F. Farias, and C. C. Moallemi. Approximate dynamic programming via a smoothed approximate linear program. *Operations Research*, Forthcoming, 2011.

A. K. Dixit and R. S. Pindyck. *Investment Under Uncertainty.* Princeton University Press, Princeton, New Jersey, USA, 1994.

A. Eydeland and K. Wolyniec. *Energy and Power Risk Management.* John Wiley and Sons Inc., Hoboken, NJ, USA, 2003.

V. F. Farias and B. Van Roy. An approximate dynamic programming approach to network revenue management. Working paper, Stanford Univ., 2007.

B.J. Felix and C. Weber. Gas storage valuation applying numerically constructed recombining trees. *European Journal of Operational Research*, 216(1):178–187, 2012.

H. Geman. *Commodities and Commodity Derivatives: Modelling and Pricing for Agriculturals, Metals, and Energy.* Wiley, Chichester, UK, 2005.

P. Glasserman. *Monte Carlo Methods in Financial Engineering.* Springer, New York, NY, USA, 2004.

F. Glover. Surrogate constraints. *Operations Research*, 16(4):741–749, 1968.

F. Glover. Surrogate constraint duality in mathematical programming. *Operations Research*, 23(3):434–451, 1975.

J. Gray and P. Khandelwal. Towards a realistic gas storage model. *Commodities Now*, pages 1–4, June 2004.

M. Griffin. Personal communication. 2011.

M. B. Haugh and L. Kogan. Pricing American options: A duality approach. *Operations Research*, 52(2): 258–270, 2004.

G. Lai, F. Margot, and N. Secomandi. An approximate dynamic programming approach to benchmark practice-based heuristics for natural gas storage valuation. *Operations Research*, 58(3):564–582, 2010.

G. Levy. *Computational Finance: Numerical Methods for Pricing Financial Instruments.* Butterworth-Heinemann, Oxford, UK, 2004.

F. A. Longstaff and E. S. Schwartz. Valuing American options by simulation: A simple least-squares approach. *Review of Financial Studies*, 14(1):113–147, 2001.

S. Maragos. Valuation of the operational flexibility of natural gas storage reservoirs. In E. Ronn, editor, *Real*

Options and Energy Management Using Options Methodology to Enhance Capital Budgeting Decisions, pages 431–456. Risk Publications, London, UK, 2002.

W. Margrabe. The value of an option to exchange one asset for another. *The Journal of Finance*, 33(1): 177–186, 1978.

C. C. Moallemi, S. Kumar, and B. Van Roy. Approximate and data-driven dynamic programming for queuing networks. Working paper, Stanford Univ., 2008.

J. R. Morrison and P. R. Kumar. New linear program performance bounds for queuing networks. *Journal of Optimization Theory and Applications*, 100(3):575–597, 1999.

M. Petrik and S. Zilberstein. Constraint relaxation in approximate linear programs. In *Proceedings of the Twenty-Sixth International Conference on Machine Learning*, pages 809–816, Montreal, Canada, 2009.

W. B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality, 2nd Edition.* John Wiley & Sons, Hoboken, New Jersey, USA, 2011.

M.L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* John Wiley & Sons, Inc., New York, NY, USA, 1994.

L. C. G. Rogers. Monte Carlo valuation of American options. *Mathematical Finance*, 12(3):271–286, 2002.

M. Rubinstein. Return to OZ. *Risk Magazine*, 1994.

P. J. Schweitzer and A. Seidmann. Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, 110(2):568–582, 1985.

N. Secomandi. Optimal commodity trading with a capacitated storage asset. *Management Science*, 56(3): 1090–1049, 2010.

N. Secomandi. The role of price spreads and reoptimization in the real option management of commodity storage assets. Working paper, Carnegie Mellon Univ., 2011.

N. Secomandi, G. Lai, F. Margot, A. Scheller-Wolf, and D. Seppi. Valuation and hedging of commodity storage in the presence of term structure model error. Working paper, Carnegie Mellon Univ., 2011.

J.E. Smith and K.F. McCardle. Options in the real world: Lessons learned in evaluating oil and gas investments. *Operations Research*, 47(1):1–15, 1999.

M. Thompson, M. Davison, and H. Rasmussen. Natural gas storage valuation and optimization: A real options application. *Naval Research Logistics*, 56(3):226–238, 2009.

D.M. Topkis. *Supermodularity and Complementarity.* Princeton University Press, Princeton, New Jersey, USA, 1998.

M. A. Trick and S. E. Zin. Spline approximations to value functions. *Macroeconomic Dynamics*, 1(1):255–277, 1997.

L. Trigeorgis. *Real Options: Managerial Flexibility and Strategy in Resource Allocation*. The MIT Press, Cambridge, MA, USA, 1996.

J.N. Tsitsiklis and B. Van Roy. Regression methods for pricing complex American-style options. *IEEE Transactions on Neural Networks*, 12(4):694–703, 2001.

B. Van Roy. *Neuro-Dynamic Programming: Overview and Recent Trends*. Handbook of Markov Decision Processes. Kluwer, Boston, MA, USA, 2002.

M. H. Veatch. Approximate linear programming for networks: Average cost bounds. Working paper, Gordon College, 2010.

O.Q. Wu, D.D. Wang, and Z. Qin. Seasonal energy storage operations with limited flexibility. Working paper, Univ. of Michigan, 2010.

D. Zhang and D. Adelman. An approximate dynamic programming approach to network revenue management with customer choice. *Transportation Science*, 43(3):381–394, 2009.

Online Appendix

A Another PSR-based ADP

Another PSR of OALP can be derived by using the same partition sets used to derive ADP1 (see §5 for details). However, we use the probabilities $\Pr(F_{i,i+1}|s_i, F_{0,i+1})$ as multipliers. That is, for a partition set identified by (i, x_i, a, s_i) we multiply each constraint corresponding to an element $(i, x_i, a, s_i, F_{i,i+1})$ in the partition by $\Pr(F_{i,i+1}|s_i, F_{0,i+1})$. Similar to ADP1, we apply this PSR of OALP and obtain the ADP

$$\phi_i(x_i, s_i) = \max_{a \in \mathcal{A}^D(x_i)} r(a, s_i) + \delta \mathbb{E}\left[\mathbb{E}\left[\phi_{i+1}(x_i - a, \tilde{s}_{i+1})|\tilde{F}_{i,i+}\right]|s_i, F_{0,i+1}\right], \tag{35}$$

$\forall i \in \mathcal{I}$ and $(x_i, s_i) \in \mathcal{X}^D \times \mathcal{F}^D_{i,i}$, with $\phi_N(x_N, s_N) = 0$, $\forall x_N \in \mathcal{X}$. This ADP differs from ADP1, ADP2, and SADP.

B Crude Oil Forward Curves and Calibration Results

This section includes the price information for our crude oil instances. Tables 2-3 report the NYMEX forward curves; Tables 4 and 5-7 report the volatilities and the correlation matrix, estimated on the NYMEX closing crude oil futures prices observed from 2002 to 2006.

Table 2: Crude oil forward curves ($/bbl) on the first trading day of each month from January to June 2006.

Months to Maturity	January	February	March	April	May	June
0	63.11	64.71	62.01	66.07	73.75	70.11
1	63.14	66.56	61.97	66.74	73.70	70.34
2	63.99	67.42	63.62	68.04	75.22	71.23
3	64.48	68.02	64.66	68.79	75.94	71.86
4	64.84	68.47	65.43	69.20	76.40	72.37
5	65.14	68.81	66.02	69.42	76.70	72.77
6	65.38	69.06	66.49	69.56	76.90	73.08
7	65.59	69.24	66.88	69.66	77.00	73.29
8	65.77	69.37	67.18	69.74	77.01	73.42
9	65.91	69.47	67.43	69.79	76.97	73.49
10	66.02	69.55	67.63	69.83	76.91	73.50
11	66.10	69.61	67.82	69.85	76.81	73.46
12	66.17	69.62	67.96	69.86	76.69	73.38
13	66.22	69.61	68.08	69.83	76.54	73.26
14	66.26	69.59	68.15	69.80	76.36	73.13
15	66.29	69.55	68.20	69.74	76.20	72.99
16	66.30	69.50	68.21	69.68	76.02	72.84
17	66.29	69.45	68.22	69.60	75.84	72.66
18	66.24	69.37	68.22	69.52	75.66	72.48
19	66.15	69.27	68.20	69.43	75.45	72.27
20	66.02	69.15	68.16	69.34	75.23	72.06
21	65.88	69.00	68.11	69.22	75.01	71.86
22	65.73	68.85	68.05	69.11	74.78	71.63
23	65.56	68.71	68.00	68.99	74.55	71.40

Table 3: NYMEX Crude oil forward curves ($/bbl) on the first trading day of each month from July to December 2006.

Months to Maturity	July	August	September	October	November	December
0	75.20	74.93	69.24	60.96	58.64	63.43
1	75.19	74.91	69.19	61.03	58.71	63.43
2	76.16	76.18	70.37	62.32	60.55	65.01
3	76.79	76.95	71.21	63.36	61.84	66.07
4	77.23	77.51	71.94	64.23	62.84	66.83
5	77.57	77.97	72.54	64.95	63.64	67.46
6	77.80	78.32	73.03	65.55	64.28	67.99
7	77.93	78.57	73.42	66.07	64.82	68.43
8	78.00	78.75	73.74	66.51	65.29	68.83
9	77.99	78.85	73.97	66.88	65.69	69.16
10	77.94	78.88	74.15	67.18	66.04	69.44
11	77.84	78.86	74.28	67.42	66.35	69.68
12	77.72	78.82	74.36	67.61	66.61	69.89
13	77.59	78.76	74.41	67.77	66.82	70.08
14	77.46	78.66	74.44	67.89	67.01	70.23
15	77.32	78.54	74.44	67.97	67.16	70.34
16	77.16	78.42	74.40	68.04	67.29	70.42
17	77.00	78.26	74.35	68.07	67.39	70.49
18	76.80	78.09	74.28	68.10	67.48	70.54
19	76.59	77.92	74.18	68.10	67.55	70.57
20	76.38	77.75	74.06	68.08	67.60	70.57
21	76.17	77.56	73.94	68.02	67.64	70.57
22	75.96	77.36	73.77	67.96	67.67	70.54
23	75.75	77.17	73.61	67.88	67.66	70.51

Table 4: Estimated crude oil futures price volatilities.

Months to Maturity	Volatility
1	0.343
2	0.317
3	0.298
4	0.284
5	0.272
6	0.263
7	0.255
8	0.248
9	0.242
10	0.236
11	0.231
12	0.227
13	0.223
14	0.220
15	0.216
16	0.213
17	0.210
18	0.208
19	0.206
20	0.204
21	0.202
22	0.202
23	0.199

Table 5: Estimated crude oil futures price correlation submatrix 1.

	Maturity											
Maturity	1	2	3	4	5	6	7	8	9	10	11	12
1	1.000	0.974	0.963	0.954	0.944	0.934	0.925	0.916	0.908	0.900	0.892	0.883
2		1.000	0.995	0.989	0.982	0.974	0.966	0.958	0.951	0.944	0.936	0.928
3			1.000	0.998	0.993	0.988	0.982	0.975	0.969	0.963	0.956	0.948
4				1.000	0.999	0.995	0.991	0.986	0.980	0.975	0.969	0.963
5					1.000	0.999	0.996	0.992	0.988	0.984	0.979	0.973
6						1.000	0.999	0.996	0.994	0.990	0.986	0.982
7							1.000	0.999	0.997	0.995	0.992	0.988
8								1.000	0.999	0.998	0.995	0.992
9									1.000	0.999	0.998	0.996
10										1.000	0.999	0.998
11											1.000	1.000
12												1.000

Table 6: Estimated crude oil futures price correlation submatrix 2.

	Maturity										
Maturity	13	14	15	16	17	18	19	20	21	22	23
1	0.873	0.865	0.855	0.847	0.837	0.828	0.818	0.808	0.799	0.785	0.782
2	0.919	0.911	0.902	0.893	0.884	0.875	0.865	0.856	0.847	0.832	0.831
3	0.940	0.932	0.924	0.916	0.907	0.899	0.889	0.880	0.872	0.857	0.856
4	0.956	0.948	0.941	0.934	0.926	0.918	0.909	0.900	0.892	0.878	0.878
5	0.967	0.961	0.954	0.947	0.940	0.933	0.925	0.917	0.909	0.895	0.896
6	0.976	0.971	0.965	0.959	0.953	0.946	0.938	0.931	0.924	0.910	0.911
7	0.983	0.979	0.973	0.968	0.963	0.956	0.950	0.943	0.936	0.923	0.924
8	0.989	0.985	0.980	0.976	0.970	0.965	0.959	0.952	0.946	0.933	0.935
9	0.993	0.989	0.985	0.981	0.977	0.972	0.966	0.960	0.955	0.942	0.944
10	0.996	0.993	0.990	0.986	0.982	0.978	0.973	0.967	0.962	0.950	0.953
11	0.998	0.996	0.993	0.991	0.987	0.983	0.979	0.974	0.969	0.957	0.960
12	0.999	0.998	0.996	0.994	0.991	0.988	0.984	0.979	0.975	0.963	0.967

Table 7: Estimated crude oil futures price correlation submatrix 3.

	Maturity										
Maturity	13	14	15	16	17	18	19	20	21	22	23
13	1.000	0.999	0.998	0.997	0.994	0.991	0.988	0.984	0.981	0.969	0.973
14		1.000	0.999	0.998	0.997	0.994	0.992	0.988	0.985	0.974	0.979
15			1.000	1.000	0.998	0.997	0.995	0.992	0.989	0.978	0.983
16				1.000	1.000	0.998	0.997	0.995	0.992	0.981	0.987
17					1.000	1.000	0.998	0.997	0.995	0.984	0.991
18						1.000	1.000	0.998	0.997	0.987	0.993
19							1.000	1.000	0.999	0.989	0.996
20								1.000	1.000	0.990	0.998
21									1.000	0.991	0.999
22										1.000	0.991
23											1.000

C Detailed Computational Results

This section reports tables with the detailed numerical results of our estimated lower and upper bounds with SADP, ADP1, and ADP2, and the CPU times required to compute them. These results supplement the discussion in §8. The results for the natural gas and crude oil instances are reported in §C.1 and §C.2, respectively.

C.1 Results for the Natural Gas Instances

This subsection contains the following tables with the numerical results for the natural gas instances (the labels of these instances are from LMS):

- Table 8 reports UB2.
- Table 9 compares the lower bound estimates LBS, LB1, and LB2 and the dual upper bound estimates UBS, UB1, and UB2.
- Table 10 compares the reoptimized greedy lower bound estimates RLBS, RLB1, and RLB2.
- Table 11 reports the overall CPU seconds for SADP, ADP1, and ADP2 (the "no reoptimization" part of this table), and the CPU seconds required to estimate the reoptimized greedy lower bounds RLBS, RLB1, and RLB2 (the "reoptimization" part of this table).

Table 8: UB2 values for the natural gas instances.

Instance	UB2
24-Sp-1	4.20
24-Sp-2	5.26
24-Sp-3	5.72
24-Su-1	4.70
24-Su-2	6.26
24-Su-3	6.78
24-Fa-1	4.14
24-Fa-2	6.38
24-Fa-3	7.50
24-Wi-1	1.80
24-Wi-2	2.48
24-Wi-3	2.85

Table 9: LBS, LB1, LB2, UBS, and UB1 normalized by UB2 for the natural gas instances.

Instance	LBS/UB2 %	LB1/UB2 %	LB2/UB2 %	UBS/UB2 %	UB1/UB2 %
24-Sp-1	94.23	93.89	96.97	101.20	101.32
24-Sp-2	95.35	95.38	98.72	101.43	101.44
24-Sp-3	96.26	96.43	99.17	101.20	101.14
24-Su-1	94.99	94.52	97.35	101.18	101.28
24-Su-2	97.11	97.29	99.40	101.02	101.04
24-Su-3	97.72	98.19	99.72	100.87	100.87
24-Fa-1	95.81	95.72	98.46	101.23	101.37
24-Fa-2	97.39	97.49	99.94	100.98	101.03
24-Fa-3	98.12	98.29	100.12	100.77	100.78
24-Wi-1	82.54	80.11	90.97	102.97	103.02
24-Wi-2	88.56	87.35	96.69	102.81	102.79
24-Wi-3	91.50	90.28	98.22	102.70	102.61

Table 10: RLBS, RLB1, and RLB2 normalized by UB2 for the natural gas instances.

Instance	RLBS/UB2 %	RLB1/UB2 %	RLB2/UB2 %
24-Sp-1	99.55	99.55	99.57
24-Sp-2	99.61	99.61	99.46
24-Sp-3	99.70	99.67	99.59
24-Su-1	99.77	99.75	99.75
24-Su-2	100.32	100.34	100.33
24-Su-3	100.26	100.26	100.20
24-Fa-1	100.18	100.16	100.10
24-Fa-2	100.64	100.64	100.64
24-Fa-3	100.40	100.36	100.38
24-Wi-1	96.49	97.42	97.62
24-Wi-2	99.12	98.32	99.25
24-Wi-3	99.04	98.53	99.10

Table 11: CPU seconds for SADP, ADP1, and ADP2 for the natural gas instances.

	No Reoptimization			Reoptimization		
Instance	SADP	ADP1	ADP2	SADP	ADP1	ADP2
24-Sp-1	308.87	10.23	168.58	596.44	92.29	1247.53
24-Sp-2	303.26	14.50	210.09	558.13	90.66	1228.55
24-Sp-3	298.54	17.16	223.33	543.59	89.94	1230.67
24-Su-1	278.67	10.21	162.23	585.07	91.98	1237.97
24-Su-2	273.26	14.51	198.65	556.09	90.69	1233.16
24-Su-3	271.83	17.11	219.73	561.30	89.78	1226.46
24-Fa-1	292.68	10.21	153.73	610.73	92.60	1233.5
24-Fa-2	294.43	14.54	186.92	614.64	90.54	1233.6
24-Fa-3	300.79	17.15	208.85	566.34	90.02	1222.4
24-Wi-1	297.97	10.24	169.38	619.23	92.12	1232.90
24-Wi-2	313.59	14.60	205.39	578.15	90.69	1229.73
24-Wi-3	295.97	17.14	225.20	572.63	89.75	1224.72

C.2 Results for the Crude Oil Instances

This subsection contains the following tables with the numerical results for our crude oil instances (labeled Jan through Dec, which abbreviate the months January through December):

1. Table 12 reports UB1 for m_i equal to $20,000$, which we label UB1*.

2. Table 13 reports the greedy lower bound estimate LB1 and the dual upper bound estimate UB1 as a percentage of UB1*.

3. Table 14 reports the reoptimized greedy lower bound estimate RLB1 as a percentage of UB1*.

4. Table 15 reports the overall CPU seconds for ADP1 when m_i equals 500, 1,000, 10,000, and 20,000, respectively (the "no reoptimization" part of this table), and the CPU seconds required to estimate the reoptimized greedy lower bound RLB1 (the "reoptimization" part of this table).

Table 12: UB1* for the crude oil instances.

Instance	UB1*
Jan	5.53
Feb	7.29
Mar	7.65
Apr	6.32
May	6.09
Jun	5.72
Jul	5.82
Aug	6.63
Sep	7.17
Oct	7.96
Nov	9.19
Dec	8.17

Table 13: LB1 for m_i equal to 500, 1000, 10,000, and 20,000, and UB1 m_i equal to 500, 1000, 10,000 normalized by UB1* for the crude oil instances.

	LB1/UB1*%				UB1/UB1*%		
	m_i				m_i		
Instance	500	1,000	10,000	20,000	500	1,000	10,000
Jan	98.34	97.81	98.92	99.63	114.53	107.64	100.34
Feb	98.73	98.05	99.25	99.15	111.37	105.83	100.26
Mar	99.43	99.50	99.37	99.92	110.64	105.19	100.18
Apr	97.78	98.39	99.02	98.89	112.87	106.44	100.20
May	98.49	98.54	99.49	99.63	113.80	106.78	100.23
Jun	98.81	98.22	99.37	99.56	114.49	107.40	100.31
Jul	98.92	98.43	99.44	99.63	114.98	107.79	100.33
Aug	98.63	98.35	99.58	99.51	113.56	107.00	100.29
Sep	98.71	98.21	99.27	99.74	112.30	106.25	100.23
Oct	99.34	99.70	99.78	100.14	110.27	105.19	100.22
Nov	99.12	99.90	99.95	99.96	108.81	104.50	100.21
Dec	98.98	99.48	99.53	100.00	110.48	105.23	100.24

Table 14: RLB1 normalized by UB1* for the crude oil instances.

Instance	RLB1/UB1* %
Jan	99.46
Feb	99.39
Mar	99.80
Apr	99.32
May	99.28
Jun	99.58
Jul	99.52
Aug	99.43
Sep	99.86
Oct	100.07
Nov	100.02
Dec	100.09

Table 15: CPU seconds for ADP1 for the crude oil instances.

	No Reoptimization				
	m_i				
Instance	500	1,000	10,000	20,000	Reoptimization
Jan	6.13	6.25	6.82	7.05	28.19
Feb	6.15	6.40	6.85	7.07	28.29
Mar	6.15	6.32	6.79	7.11	28.40
Apr	6.16	6.30	6.78	7.07	28.04
May	6.17	6.26	6.84	7.11	28.17
Jun	6.11	6.34	6.72	7.02	28.16
Jul	6.21	6.30	6.75	7.07	28.23
Aug	6.12	6.40	6.86	7.10	28.02
Sep	6.11	6.31	6.76	7.07	28.38
Oct	6.10	6.29	6.81	7.07	28.68
Nov	6.10	6.29	6.79	7.04	28.39
Dec	6.22	6.26	6.88	7.08	28.52