Operations Research Letters 45 (2017) 409–414

Contents lists available at ScienceDirect

Operations Research Letters

journal homepage: www.elsevier.com/locate/orl

Relationship between least squares Monte Carlo and approximate linear programming[☆]

Selvaprabu Nadarajah [a,*], Nicola Secomandi [b]

[a] College of Business Administration, University of Illinois at Chicago, 601 South Morgan Street, Chicago, IL, 60607, USA

[b] Tepper School of Business, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA

ARTICLE INFO

Article history:
Received 3 August 2016
Received in revised form 29 May 2017
Accepted 29 May 2017
Available online 15 June 2017

Keywords:
Markov decision processes
Approximate dynamic programming
Least squares Monte Carlo
Approximate linear programming
Financial and real options
Energy storage

ABSTRACT

Least squares Monte Carlo (LSM) is commonly used to manage and value early or multiple exercise financial or real options. Recent research in this area has started applying approximate linear programming (ALP) and its relaxations, which aim at addressing a possible ALP drawback. We show that regress-later LSM is itself an ALP relaxation that potentially corrects this ALP shortcoming. Our analysis consolidates two streams of research and supports using this LSM version rather than ALP on the considered models.

1. Introduction

The optimal exercise and valuation of options play an important role in financial engineering and real option applications. Examples include American/Bermudan options [23], [30, chapter 8], swing options [6,16,35,40,44], [49, chapter 20], [53, chapter 7], switching options [18,26,56], and energy storage assets modeled as real options [5,10,13,32,37,43,44,51,57]. The valuation of these types of options typically gives rise to intractable Markov decision processes (MDPs).

The extant literature commonly approaches the solution of these MDPs using the least squares Monte Carlo (LSM) approach [5,9–11,13,15,18,21,26,27,29,30], [31, chapter 8], [34, chapter 7], [38,44,45,52,55,61]. LSM computes a value/continuation function approximation, which can then be used to obtain a feasible option exercise policy and estimate a lower bound. LSM has two basic variants: (i) the standard regress-now version (LSMN) that approximates the MDP continuation function [14], [30, chapter 8], [38,61], and (ii) the regress-later version (LSML) that approximates the MDP value function [9,11,31,44]. Whereas LSMN is commonly used, Beutner et al. [9] and Nadarajah et al. [44] provide theoretical and numerical support for using LSML instead.

Recently, Christensen [17] and Nadarajah et al. [43] have applied the approximate linear programming (ALP) approach [50] to American option and commodity storage valuation problems, respectively. ALP is an approximate dynamic programming (ADP) approach mainly used in economics and operations research applications (see, e.g., [1–4,19,22,24,25,33,36,41,42,58–60,62,63]). Despite these successful applications of ALP and its analogous use by Christensen [17], Nadarajah et al. [43] finds that it can perform poorly for a particular class of value function approximations (VFAs). Moreover, they establish that formulating and solving ALP relaxations yields near optimal policies. These authors attribute the dismal observed ALP performance to a possible distortion between optimal ALP dual solutions and stage-state-action visit frequencies of optimal MDP deterministic policies. Petrik and Zilberstein [46] and Desai et al. [22] identify similar issues with ALP and propose alternative ALP relaxations to overcome them.

In this paper we show that LSML is itself an ALP relaxation. This result consolidates two separate streams of research. It is based on a novel application of the ALP relaxation framework proposed in [43]. This scheme relies on adding constraints to the ALP dual and taking the dual of this restricted model to obtain an ALP relaxation. The ALP dual restricting constraints that we use and the resulting ALP relaxation differ from the ones available in the existing literature [22,43,46]. Our analysis suggests that these constraints might alleviate the stated potential ALP drawback. LSML is both simpler (easier to code) and more practical (faster to execute) than ALP. Our finding adds potentially improved accuracy to these advantages of using LSML rather than ALP to

☆ The previous version of this paper was titled "Regress-later Least Squares Monte Carlo: Duality Perspective and Energy Real Option Application".

* Corresponding author.
E-mail addresses: selvan@uic.edu (S. Nadarajah), ns7@andrew.cmu.edu (N. Secomandi).

http://dx.doi.org/10.1016/j.orl.2017.05.010
0167-6377/

approximately solve intractable MDPs that arise in financial engineering and real option applications. We apply LSML and ALP to a set of natural gas storage instances on which LSML is known to lead to near optimal operating policies and almost tight (dual) upper bounds on the value of an optimal policy [44]. We observe that LSML yields substantially better storage policies than ALP, in addition to exhibiting considerably shorter run times. These results complement the numerical comparison of LSMN and ALP performed in [43] in the same context. Moreover, although our primary focus is on LSML, we establish that LSMN is a relaxation of a linear program related to the ALP model, which we formulate.

In Section 2 we formulate an MDP for the management and valuation of options with early and multiple exercise. We present LSML in Section 3. We apply ALP to our MDP in Section 4. In Section 5 we establish that LSML is an ALP relaxation. We briefly summarize our numerical findings in Section 6. We conclude in Section 7. An online supplement includes proofs, the details of our numerical study, and the LSMN analysis.

2. MDP

In this section we formulate a finite horizon MDP for the valuation and management of options that feature early and multiple exercise opportunities. There are I stages that belong to the set $\mathcal{I} := \{0, \dots, I-1\}$ and are indexed by i. Each stage i corresponds to an option exercise date T_i. The status of the option is denoted by the scalar x_i and belongs to the discrete set $\mathcal{X}_i$. The initial option status is x_0, that is, $\mathcal{X}_0 := \{x_0\}$. The information state is the vector $w_i \in \mathcal{W}_i \subseteq \mathbb{R}^{I-i}$. It represents, for example, the option underlying asset term structure $(w_{i,i}, w_{i,i+1}, \dots, w_{i,I-1})$, where $w_{i,j}$ is the element of the term structure corresponding to date T_j at time T_i. For instance, $w_{i,i}$ might be the spot price of an asset and $w_{i,j}$ might be the date T_i price of a futures contract for this asset with delivery at time T_j. We assume that each set $\mathcal{W}_i$ is discrete. Our assumption simply avoids technical complications in our theoretical analysis performed in Section 5. We deviate from this assumption in our numerical investigation summarized in Section 6. The stage i state space is $\mathcal{X}_i \times \mathcal{W}_i$. We let $\mathcal{X}_I$ be the terminal option status set. We define $\mathcal{W}_I$ as the singleton $\{0\}$.

The option exercise action a_i at stage i and state (x_i, w_i) belongs to the discrete set $\mathcal{A}_i(x_i)$. Performing this action results in an immediate reward given by the function $r_i(x_i, w_i, a_i)$. The transition function $f_i(x_i, a_i)$ specifies the corresponding stage $i+1$ option status. The information state evolves from w_i to w_{i+1} according to a known stochastic process independent of x_i and a_i. We discount cash flows using the per-stage risk-free discount factor $\delta \in (0, 1]$. There is no reward associated with any terminal option status.

Let Π define the set of all feasible policies. We denote by A_i^π the stage i decision rule of feasible policy π applied to state (x_i, w_i). An optimal policy solves

$$\max_{\pi \in \Pi} \sum_{i \in \mathcal{I}} \delta^i \mathbb{E}[r_i(x_i^\pi, w_i, A_i^\pi(x_i^\pi, w_i)) | x_0, w_0],$$

where $\mathbb{E}$ denotes expectation under a risk neutral measure [54] for the stochastic process that describes the evolution of the information state, and x_i^π is the random option status at stage i under policy π.

In theory, an optimal set of actions for each stage and state can be obtained by solving the following stochastic dynamic program (SDP) with boundary conditions $V_I(x_I, w_I) := 0$ for each $(x_I, w_I) \in \mathcal{X}_I \times \mathcal{W}_I$:

$$V_i(x_i, w_i) = \max_{a_i \in \mathcal{A}_i(x_i)} \{r_i(x_i, w_i, a_i) + \delta \mathbb{E}[V_{i+1}(f_i(x_i, a_i), w_{i+1}) | w_i]\}, \quad (1)$$

for each $(i, x_i, w_i) \in \mathcal{I} \times \mathcal{X}_i \times \mathcal{W}_i$, where $V_i(x_i, w_i)$ denotes the stage i value function at state (x_i, w_i). Nadarajah et al. [44] illustrates how a model that closely resembles SDP (1) can be specified to represent swing and storage options, which generalize simpler options of the Bermudan type (see, e.g., [23] and [30, chapter 8]).

To ease the exposition, for the most part in the rest of this paper we write (i, x_i, w_i, a_i) instead of $(i, x_i, w_i, a_i) \in \mathcal{I} \times \mathcal{X}_i \times \mathcal{W}_i \times \mathcal{A}_i(x_i)$, and use $(\cdot)_{-(i)}$ to indicate that i is excluded from $\mathcal{I}$ in the tuple ground set.

3. LSML

In this section we present the LSML method for our MDP. This material is based on [44, §3.2]. LSML approximates the value function of SDP (1). Specifically, it expresses the stage i VFA as a linear combination of a given set of *basis* functions, which is a common approach in ADP (see, e.g., [7, chapter 6.1.1], [28], [30, p. 430], and [47, p. 326]). The bth basis function at stage i is $\phi_{i,b}$. The weight associated with this basis function when the option status is x_i is $\beta_{i,x_i,b}$. We define the set of these basis functions and the vector of their associated weights by $\phi_i := \{\phi_{i,b}, b \in \mathcal{B}_i\}$ and $\beta_{i,x_i} := (\beta_{i,x_i,b}, b \in \mathcal{B}_i)$, respectively, where $\mathcal{B}_i := \{1, 2, \dots, B_i\}$ and B_i is a positive integer. The stage i VFA at state (x_i, w_i) is $\sum_{b \in \mathcal{B}_i} \phi_{i,b}(w_i)\beta_{i,x_i,b}$. In each stage, our VFA relies on basis functions that depend only on the information state, whereas the option status is an index of the weights. This modeling approach is common in the LSM literature (see, e.g., [10,18,26,38,44]). However, the methods and analysis discussed in this paper can be extended to the case when the option status is an argument of the basis functions rather than an index of their weights.

Algorithm 1: LSML

1. Generate the set $\{w_i^p, (i, p) \in \mathcal{I} \times \mathcal{P}\}$ of sample paths of the information state and define $\mathcal{W}_i^{\mathcal{P}} := \{w_i^p, p \in \mathcal{P}\}$ for each stage $i \in \mathcal{I}$.
2. Set $\beta_{I,x_I}^{\text{LSM}}$ to 0 for each $x_I \in \mathcal{X}_I$.
3. For each $i = I-1$ to 0 do:
   For each $x_i \in \mathcal{X}_i$ do:
   (i) For each $p \in \mathcal{P}$ do: compute the value function estimate
   $$v_i(x_i, w_i^p) := \max_{a_i \in \mathcal{A}_i(x_i)} \Bigg\{ r_i(x_i, w_i^p, a_i) + \delta \sum_{b \in \mathcal{B}_{i+1}} \mathbb{E}[\phi_{i+1,b}(w_{i+1}) | w_i^p] \beta_{i+1, f_i(x_i,a_i), b}^{\text{LSM}} \Bigg\}. \quad (2)$$
   (ii) $\beta_{i,x_i}^{\text{LSM}} := \min_{\beta_{i,x_i} \in \mathbb{R}^{B_i}} \| \sum_{b \in \mathcal{B}_i} \phi_{i,b}(\cdot)\beta_{i,x_i,b} - v_i(x_i, \cdot) \|_2$, where $\| \cdot \|_2$ denotes two-norm.
4. Return $\beta_{i,x_i}^{\text{LSM}}$ for each $(i, x_i) \in \mathcal{I} \times \mathcal{X}_i$.

Algorithm 1 summarizes the LSML steps. LSML generates the set of P information state Monte Carlo sample paths $\{w_i^p, (i, p) \in \mathcal{I} \times \mathcal{P}\}$, with $\mathcal{P} := \{1, 2, \dots, P\}$, starting from the known information state w_0 in stage 0, and uses it to define the set $\mathcal{W}_i^{\mathcal{P}} := \{w_i^p, p \in \mathcal{P}\}$ of sampled information states for each stage $i \in \mathcal{I}$. The LSML terminal condition is to set to zero the vector $\beta_{I,x_I}^{\text{LSM}}$ for each option status x_I (where the superscript LSM abbreviates least squares Monte Carlo). Beginning from stage $I-1$ and moving backward to stage zero, for each option status x_i LSML executes steps (i) and (ii). In step (i) for each index $p \in \mathcal{P}$ it computes the estimate $v_i(x_i, w_i^p)$ of the stage i value function at option status x_i and information