Learning Exercise Policies for American Options

Yuxi Li
Dept. of Computing Science
University of Alberta
Edmonton, Alberta
Canada T6G 2E8

Csaba Szepesvari
Dept. of Computing Science
University of Alberta
Edmonton, Alberta
Canada T6G 2E8

Dale Schuurmans
Dept. of Computing Science
University of Alberta
Edmonton, Alberta
Canada T6G 2E8

Abstract

Options are important instruments in modern finance. In this paper, we investigate reinforcement learning (RL) methods—in particular, least-squares policy iteration (LSPI)—for the problem of learning exercise policies for American options. We develop finite-time bounds on the performance of the policy obtained with LSPI and compare LSPI and the fitted Q-iteration algorithm (FQI) with the Longstaff-Schwartz method (LSM), the standard least-squares Monte Carlo algorithm from the finance community. Our empirical results show that the exercise policies discovered by LSPI and FQI gain larger payoffs than those discovered by LSM, on both real and synthetic data. Furthermore, we find that for all methods the policies learned from real data generally gain similar payoffs to the policies learned from simulated data. Our work shows that solution methods developed in machine learning can advance the state-of-the-art in an important and challenging application area, while demonstrating that computational finance remains a promising area for future applications of machine learning methods.

1 Introduction

A *call (put) option* gives the holder the right, not the obligation, to buy (respectively, sell) an underlying asset—for example, a share of a stock—by a certain date (the maturity date) for a certain price (the strike price). An *American option* can be exercised at any time up to the maturity date. A challenging problem that arises in connection to trading options is to determine the option's "fair" price, called *option pricing*. This problem can be solved by finding an optimal exercise policy, i.e., a feedback rule that determines when to exercise the option given a strike price so as to maximize expected total discounted return. This is an optimal stopping problem that belongs to the general class of Markov Decision Processes (MDP) (Puterman 1994; Bertsekas 1995). In theory, assuming perfect knowledge of the MDP, dynamic programming can be used to find the optimal policy (Bertsekas and Tsitsiklis 1996). When the size of an MDP is large, one encounters the "curse of dimensionality", which calls for effective approximate, sampling based techniques. In cases when the MDP is unknown and no simulator is available, only sample trajectories can be used to infer a good policy. In this paper we will look into reinforcement learning (RL) methods (Bertsekas and Tsitsiklis 1996; Sutton and Barto 1998) that are able to address these issues and study their suitability for learning exercise policies for option pricing.

This paper makes two contributions: On the one hand we derive a high-probability, finite-time bound on the performance of least-squares policy iteration (LSPI) in Lagoudakis and Parr (2003) for option pricing following the work of Antos et al. (2008). This bound complements the results of Tsitsiklis and Van Roy (2001), who showed that it is crucial to use the distribution of price trajectories in order to avoid an exponential blow-up of the approximation error with the time-horizon of the problem when considering one version of fitted Q-iteration. Here we also find that it is crucial to use this distribution and demonstrate a very mild dependence on the horizon. In comparison with the asymptotic results of Tsitsiklis and Van Roy (2001), our results make both the dependence on the approximation and estimation errors (due to the finiteness of the sample) explicit.

Appearing in Proceedings of the 12^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2009, Clearwater Beach, Florida, USA. Volume 5 of JMLR: W&CP 5.

The second contribution is an empirical comparison of LSPI, fitted Q-iteration (FQI) as proposed under the name of "approximate value iteration" by Tsitsiklis and Van Roy (2001) and the Longstaff-Schwartz method (LSM) (Longstaff and Schwartz 2001), the latter of which is a standard approach from the finance literature for pricing American options. Our results show that the RL techniques are superior to the standard approach: Exercise policies discovered by LSPI and FQI can achieve larger payoffs than those found by LSM. Furthermore, for LSPI, FQI and LSM, policies discovered based on sample paths composed directly from real data gain similar payoffs when tested on separate data to policies discovered based on sample paths generated by simulation models whose model parameters were estimated from the same dataset.

There is a vast literature on option pricing. For example, Hull (2006) provides an introduction to options and other financial derivatives and their pricing methods; Broadie and Detemple (2004) survey option pricing methods; and Glasserman (2004) provides a book length treatment for Monte Carlo methods. However, most previous work focuses on obtaining the option price, but not the exercise policy.

The remainder of this paper is organized as follows. First, we introduce basic concepts and notations to describe MDPs. Next, we present the option pricing problem, followed by the presentation of the algorithms, LSPI, FQI and LSM. We then develop finite-time bounds on the performance of the policy obtained with LSPI. After this we study empirically the performance of the algorithms, followed by our conclusions.

2 Markov decision processes

The problem of sequential decision making is common in economics, science and engineering. Many of these problems can be modeled as MDPs. A (discounted) MDP is defined by the 5-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$: $\mathcal{S}$ is a set of states; $\mathcal{A}$ is a set of actions; $\mathcal{P}$ is the transition probability kernel with $\mathcal{P}(.|s,a)$ specifying the probability of next states when the process is at state s and action a is taken; r is a reward function, with $r(s'|s,a)$ being the reward received when transitioning to state s' starting from state s and taking action a; and $\gamma \in (0,1)$ is a discount factor.

A (stationary) policy π is a rule for selecting actions based on the last state visited: $\pi(s,a)$ specifies the probability of selecting action a in state s while following policy π. We shall also use the notation $\pi(a|s)$ to denote these quantities. When $\pi(a|s) = 1$ for some action a in each state s, policy π is called deterministic. In this case we identify it with a mapping from the states to the action set.

An optimal policy maximizes the total, expected, discounted rewards obtained over the long run, for example, $\sum_{t=0}^{\infty} \gamma^t r_t$, where r_t is the reward obtained in the tth time step. The action-value function of a policy π, $Q^\pi(s,a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_0 = a\right]$, assigns to the state-action pair (s,a) the expected, discounted, total reward achieved when the initial state is s, the first action is a after which policy π is followed and where the expectation is taken with respect to the measure generated by π and P. The optimal action-value function satisfies $Q^*(s,a) = \sup_\pi Q^\pi(s,a)$. A policy that in every state s chooses an action maximizing $Q^*(s,a)$ is optimal in the sense that for any state no other policy can achieve a higher expected, discounted, total reward. Q^* can be found by either policy iteration or value iteration, at least in finite, known MDPs (Bertsekas and Tsitsiklis 1996). Value-function based reinforcement learning techniques mimic these basic algorithms to learn Q^* (or Q^π) but perform the calculations based on sampled trajectories instead of the exact calculations and employ function approximation to allow generalization to unseen states. One function approximation method that is both simple, yet powerful is to use a linear architecture, i.e., a linear combination of some basis functions: $Q(s,a;w) = \sum_{i=1}^{d} \varphi_i(s,a) w_i$, where d is the number of basis functions, $\varphi_i(\cdot,\cdot)$ is the ith basis function and $w_i \in \mathbb{R}$ is its weight, $w = (w_1, \ldots, w_d)^\mathsf{T}$. If we define $\varphi(s,a) = (\varphi_1(s,a), \ldots, \varphi_d(s,a))^\mathsf{T}$ then we can write $Q(s,a;w) = w^\mathsf{T}\varphi(s,a)$.

3 Learning exercise policies for American options

The problem of learning an exercise policy (in discrete time) can be formulated as an MDP, more specifically as a finite-horizon optimal stopping problem as follows:[1] Let T denote the number of stages which will be indexed by $t = 0, 1, 2, \ldots, T-1$. We assume that the stock price S_t at time t underlying the option evolves according to a Markov process: $S_{t+1} \sim \mathcal{P}_t(\cdot|S_t)$ $(t = 0, 1, \ldots, T-2)$. Here $S_t \in \mathcal{S}_t$, $S_0 \sim \mathcal{P}_0(\cdot)$, where $\mathcal{P}_0(\cdot)$ is a stochastic kernel over $\mathcal{S}_0$, and for $S_t \in \mathcal{S}_t$, $\mathcal{P}_t(\cdot|S_t)$ is a stochastic kernel over $\mathcal{S}_{t+1}$. Let $\mathcal{S}^* = \cup_{t=0}^{T-1} \mathcal{S}_t \times \{t\}$ and $\mathcal{S} = \{e\} \cup \mathcal{S}^*$, where e is a special state, called the *exit state*. Any f with domain $\mathcal{S}^*$ can also be viewed as a sequence $(f_0, \ldots, f_{T-1})$ of functions, where the domain of f_t is $\mathcal{S}_t$, and $f_t(S_t) = f((S_t, t))$. This equivalent representation will be used when convenient.

The exercise policy is a mapping π from $\mathcal{S}$ to the two-

[1] We discretize the time, thus, strictly speaking, the options become Bermudan. As the number of time steps approaches infinity, the solution approaches that for an American option.

element action set, $\mathcal{A} = \{0, 1\}$. (For simplicity we deal with deterministic, Markov policies only as this class of policies is sufficiently large to contain an optimal policy.) Policy $\pi = (\pi_0, \ldots, \pi_{T-1})$ determines when to exercise the option: Exercising happens when $\pi_t(S_t) = 1$. In this case the next state becomes the exit state e, which is an absorbing state. Otherwise, the next state is obtained from the stochastic kernel $\mathcal{P}_t(\cdot|S_t)$.

The rewards, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ are determined as follows: When the option is exercised at time step t in state $s \in \mathcal{S}_t$, the reward is $r((s,t),1) = g(s)$. In all other cases the reward is zero. For an American stock put options, the reward is $g(s) = \max(0, K - s)$, where $K > 0$ is the strike price. The returns are discounted with a discount factor $\gamma \in (0, 1)$.

4 Algorithms

We present the algorithms, LSPI, FQI and LSM.

4.1 LSPI for learning exercise policies

Policy iteration is a method of discovering an optimal solution for an MDP. LSPI combines the data efficiency of the least squares temporal difference (LSTD) method (Bradtke and Barto 1996) and the policy search efficiency of policy iteration. The algorithm uses a linear architecture for representing action-value functions, as described in Section 2. The linear architecture is initialized by choosing w_0 arbitrarily. In iteration $i \geq 0$, we are given $w^{(i)}$, which implicitly defines $\pi^{(i)}$, the policy that selects in each state s the action a that maximizes $Q^{(i)}(s,a) = (w^{(i)})^\mathsf{T}\varphi(s,a)$. Then LSTD is used to compute an approximation to the action-value function of this policy. This is done as follows: The data is assumed to be a trajectory (or a set of trajectories) of the form $s_1, a_1, R_1, s_2, a_2, R_2, \ldots$. Given the data, one computes $\mathbf{A}^{(i)} = \sum_t \varphi(s_t, a_t)(\varphi(s_t, a_t) - \gamma\varphi(s_{t+1}, \pi^{(i)}(s_{t+1})))^\mathsf{T}$ and $b = \sum_t \varphi(s_t, a_t)R_t$ (b can be computed at the beginning of the iterations) and then solves $\mathbf{A}^{(i)}w^{(i+1)} = b$ to obtain the next weight vector. In this way LSTD finds the solution of the TD(0) equilibrium equation $\sum_t(R_t + \gamma Q(s_{t+1}, \pi^{(i)}(s_{t+1}); w) - Q(s_t, a_t; w))\varphi(s_t, a_t) = 0$. The iteration is typically continued until the changes in the weight vector become small. The boundedness property of LSPI with respect to the L_∞-norm was established in Lagoudakis and Parr (2003). More recently, tighter finite-time bounds were obtained in Antos et al. (2008) for continuous state spaces who gave an interpretation of LSTD in terms of empirical loss minimization. Below we will derive a finite-time bound for our proposed algorithm using the techniques developed in Antos et al. (2008).

We need to consider the peculiarities of option pricing when applying LSPI to it. Since this problem is an episodic, optimal stopping problem we must incorporate time as a component in the state space. The action space has two elements: exercise (stop) and continue. The state-action value function of exercising the option, that is, the intrinsic value of the option, can be calculated exactly. Hence, we only need to learn the state-action value function for continuation. Since the states in this case are price-time pairs of the form (s,t), we let $Q((s,t),0;w) = w^\mathsf{T}\varphi(s,t)$ and let $Q((s,t),1;w) = g(s)$. Let us assume that the data consists of sample trajectories $(S_t^j; j = 1, \ldots, N, t = 0, \ldots, T-1)$, where S_t^j is the value of the tth stock price S_t in the j-th trajectory. Solving the TD(0) equilibrium equation for the weights of the redefined action-value function we get that $w^{(i+1)}$ is defined by $\mathbf{A}^{(i)}w^{(i+1)} = b^{(i)}$, where $\mathbf{A}^{(i)} = \sum_{t,j} \varphi(S_t^{(j)}, t)(\varphi(S_t^{(j)}, t) - \gamma\mathbb{I}_{\{\pi^{(i)}(S_{t+1}^{(j)})=0\}}\varphi(S_{t+1}^{(j)}, t+1))^\mathsf{T}$ and $b^{(i)} = \gamma\sum_{t,j} \varphi(S_t^{(j)}, t)\mathbb{I}_{\{\pi^{(i)}(S_{t+1}^{(j)})=1\}}g(S_{t+1}^{(j)})$. Here we exploited that the immediate rewards are always zero. Further, $\mathbb{I}_{\{L\}}$ denotes the indicator function that is 1 when the argument L is true, and otherwise it is zero.

4.2 FQI for learning exercise policies

In Section 6 of their paper (Tsitsiklis and Van Roy 2001), the authors proposed the following algorithm: Let $Q^{(i)}((s,t),0) = (w^{(i)})^\mathsf{T}\varphi(s,t)$ be the estimate of the value of the continuation action obtained in the ith step of the algorithm, where $\varphi(s,t), w \in \mathbb{R}^d$. Let $\mathbf{A}^{(i)} = \sum_{t,j} \varphi(S_t^j, t)\varphi(S_t^j, t)^\mathsf{T}$ and $b^{(i)} = \gamma\sum_{t,j} \varphi(S_t^j, t)\max(g(S_{t+1}^j), Q^{(i)}((S_{t+1}^{(j)}, t+1), 0))$. Then $w^{(i+1)} = (\mathbf{A}^{(i)})^{-1}b^{(i)}$. In fact, $w^{(i+1)}$ is the solution of the least-squares problem $\sum_{t,j}[Q((S_t^{(j)}, t), 0; w) - \gamma\max_{a\in\mathcal{A}} Q^{(i)}((S_{t+1}^{(j)}, t+1), a)]^2 \stackrel{w}{\to} \min$, where we used $Q^{(i)}((s,t),1) = g(s)$ for the sake of uniformity. Hence, this algorithm can be seen to implement a least-squares approach to fitted Q-iteration.

4.3 Least squares Monte Carlo

LSM (Longstaff and Schwartz 2001) follows the backward-recursive dynamic programming approach: Let $\varphi : \mathbb{R} \to \mathbb{R}^d$ be the basis functions used. The continuation value at stage t is estimated using $Q((s,t),0;w) = w^\mathsf{T}\varphi(s)$, except for the last stage $t = T-1$, when we set $Q((s,T-1),0;w) = g(s)$. The continuation value at stage $t < T-1$ is then estimated by finding the minimizer w_t of $L_t(w) = \sum_{j=1}^M [Q((s,t),0;w) - \max(g(S_{t+1}^j), \gamma Q((S_{t+1}^j, t+1); w_{t+1}))]^2$. Note that

this is an ordinary least squares problem in w and the method can indeed be interpreted as solving the backward-recursive dynamic programming equations in a recursive manner. While Longstaff and Schwartz (2001) obtained asymptotic results as the number of basis functions is increased, Tsitsiklis and Van Roy (2001) obtained (in the first part of their paper) asymptotic error bounds for this algorithm when the number of samples grows to infinity but the number of basis function is kept fixed.

5 Theoretical results

In this section we develop finite-time bounds on the performance of the policy obtained with LSPI. Similar bounds can be developed for other algorithms with similar arguments, which is left as future work.

Let $Q_{\max} = R_{\max}/(1-\gamma)$, where $R_{\max}$ is an upper bound on the reward function r. Let ν_t denote the distribution of prices at stage t and let $\nu = (\nu_0, \ldots, \nu_{T-1})$. We assume that the training sample consists of a number of independently sampled price trajectories (correlation between the subsequent trajectories is not considered for the sake of simplicity, but can be handled without any difficulties). Let N be the number of these trajectories. Hence, the total number of training samples is $N\,T$.

The bound will depend on the so-called total inherent Bellman-error associated with the function space $\mathcal{F} = \{\, h : \mathcal{S} \times \mathcal{A} \to \mathbb{R} \,:\, h(s,1) = r(s,1), h(s,0) = \theta^{\mathsf{T}}\varphi(s), \theta \in \mathbb{R}^d, h(e) = 0 \,\}$, where $\varphi : \mathcal{S} \to \mathbb{R}^d$ is the feature extraction method used in the procedure and e is the "exit state" of the process. Let π'_Q be the policy that is greedy w.r.t. a bounded action value function Q (in the case when multiple such policies exist, we choose one in a systematic way). Define

$$E_\infty(\mathcal{F}) = \sup_{Q' \in \mathcal{F}} \inf_{Q \in \mathcal{F}} \|Q - T^{\pi'_{Q'}} Q\|_\nu$$

and

$$E_1(\mathcal{F}) = \sup_{Q', Q'' \in \mathcal{F}} \inf_{Q \in \mathcal{F}} \|Q - T^{\pi'_{Q'}} Q''\|_\nu,$$

where $\|\cdot\|_\nu$ denotes the $L^2(\nu)$ norm. In Antos et al. (2008) the first quantity is called the inherent Bellman-error of $\mathcal{F}$, while the second is called the inherent one-step Bellman-error. They characterize the suitability of the features for representing *(i)* the fixed points of policies that can be obtained in a policy iteration procedure (the set of accessible policies) and *(ii)* the one-step Bellman-image of the available functions under the Bellman operator of accessible policies. We let $E(\mathcal{F}) = (E_\infty^2(\mathcal{F}) + E_1^2(\mathcal{F}))^{1/2}$. Following an argument in Munos and Szepesvári (2008) one can show that when the power of the feature set, φ, is increased the total inherent Bellman-error will converge to zero when the kernels $\mathcal{P}_t(\cdot|S_t)$ depend smoothly on S_t. In the result below we need $\mathcal{S}_c = \{\, (s,0) \,:\, s \in \mathcal{S} \,\} \subset \mathcal{S} \times \mathcal{A}$.

Our main result is as follows:

Theorem 1 (Bound on the Performance of LSPI for Pricing). *Execute LSPI with a training set of size $n = NT$. Consider the policy $\pi_{M,n}$ that is greedy policy with respect to the action-value function obtained by LSPI after $M > 0$ iterations. Then, for any $0 < \delta < 1$, with probability at least $1 - \delta$,*

$$\begin{aligned} & \|(Q^* - Q^{\pi_{M,n}})|_{\mathcal{S}_c}\|_\nu \\ \le\ & \frac{2\gamma}{(1-\gamma)^2}\left(E(\mathcal{F}) + \Delta_n + \gamma^{M/2} R_{\max}\right). \end{aligned}$$

Here $\Delta_n = C(\Lambda \max(\Lambda, 1)/n)^{1/4}$, where $\Lambda = d\log(n/(1-\gamma)) + \log(M/\delta) + T$ and where $C > 0$ is a universal constant.

The theorem follows by some adjustments to the proof of Theorem 4 of Antos et al. (2008) and some calculations where we estimate constants in the bound of Theorem 4. In the following, we explain the bound in the result. We give highlights of the proof in the Appendix.

First, note that when $n \to \infty$,

$$\|(Q^* - Q^{\pi_{M,\infty}})|_{\mathcal{S}_c}\|_\nu \le \frac{2\gamma}{(1-\gamma)^2}\left(E(\mathcal{F}) + \gamma^{M/2} R_{\max}\right).$$

Here we see the factor $2\gamma/(1-\gamma)^2$ that is usual in results that use contraction arguments to bound the *performance* of a policy that is greedy with respect to an approximate value function (see Bertsekas and Tsitsiklis (1996)). The term $E(\mathcal{F})$, as explained above, depends only on the features and bounds the approximation error. The two terms contributing to $E(\mathcal{F})$ arise from rewriting the LSTD fixed point equation in terms of an empirical loss function that has two corresponding terms. The supremum over Q' in the definition of these quantities arises because after the first iteration all we can know about the policy to be evaluated is that it is greedy w.r.t. some function in our chosen function set. The reason for the supremum over Q'' is similar in the definition of $E_1^2(\mathcal{F})$. Finally, $\gamma^{M/2} R_{\max}$ bounds the error due to running the algorithm only for a finite number of iterations. We also see that the rate of convergence as a function of the number of samples is $n^{-1/4}$, which, considering the squared-error, gives the familiar rate $n^{-1/2}$. The nice property of this bound compared to the general bound of Antos et al. (2008) is that in their bound a so-called concentration coefficient (denoted there by

$C_{\rho,\nu}$) appears in a multiplicative manner. This quantity depends on the MDP and is typically difficult to control. The role of this quantity is to bound the error resulting from changing measures, which arises since typically the distribution of the training samples is different from both the "test" distribution and the stationary distributions of the policies encountered during the iterations of the algorithm. Interestingly, in our case we are able to bound this constant by 1. This is because the special nature of stopping problems: Consider a stochastic policy π and the distribution μ_π defined over the state-action pairs that is obtained for a given t by sampling the prices from their underlying distribution (not caring about if π could have stopped before) and then sampling the actions from π. Imagine that we follow the process for one step from μ_π (thus the action probabilities come from π) and upon reaching the next state we generate actions at random from some other stochastic policy π'. Let $\mu_{\pi,\pi'}$ be the resulting distribution. Then, since π either exits or stays in, i.e., it can only decrease the probability masses carried to the next stage, the marginal of $\mu_{\pi,\pi'}$ on the price-time pairs can be upper bounded by the corresponding marginal of $\mu_{\pi'}$. Hence, if the distribution used to sample the prices and to evaluate the performance of the policy are the same, as in all the procedures described above, the concentration coefficient can be upper bounded by 1.

Note that the result, just like the one for the LSM algorithm by Tsitsiklis and Van Roy (2001) bounds the L^2 error. However, our bound is for the value function of the policy, while the bound in Tsitsiklis and Van Roy (2001) is for the action-value function estimate. Further, the bound in Tsitsiklis and Van Roy (2001) considers only the approximation error (i.e., when $n \to \infty$) and the proof is considerably simpler given that the solutions in LSM are defined with a backward recursion. In fact, the bounding technique in Tsitsiklis and Van Roy (2001) can possibly used to bound $E(\mathcal{F})$.

6 Empirical study

We study empirically the performance of LSPI, FQI and LSM. We study the plain American put stock options. We focus on at-the-money options, that is, the strike price is equal to the initial stock price. For simplicity, we assume the risk-free interest rate r is constant and stocks are non-dividend-paying. We assume 252 trading days in each year. We study options with quarterly, semi-annual and annual maturity terms, with 63, 126 and 252 days duration respectively. Each time step is one trading day, that is, 1/252 trading year. We set the discount factor to $\gamma = e^{-r/252}$, corresponding to the daily interest rate. LSPI and FQI iterate on the sample paths until the difference between two successive policies is sufficiently small, or when it has run 15 iterations (LSPI and FQI usually converge in 4 or 5 iterations). We obtain five years' daily stock prices from January 2002 to December 2006 for Dow Jones 30 companies from WRDS, Wharton Research Data Services.

6.1 Simulation models

In our experiments when a simulation model is used, synthetic data may be generated from either a geometric Brownian motion (GBM) model or a generalized autoregressive conditional heteroskedasticity (GARCH) model, two of the most widely used models for stock price movement (Hull 2006).

Geometric Brownian motion model. Suppose S_t, the stock price at time t, follows a GBM: $dS_t = \mu S_t dt + \sigma S_t dW_t$, where μ is the risk-neutral expected stock return, σ is the stock volatility and W is a standard Brownian motion. For a non-dividend-paying stock, $\mu = r$, the risk-free interest rate. It is usually more accurate to simulate $\ln S_t$ in practice. Using Itô's lemma, the process followed by $\ln S_t$ is: $d\ln S_t = (\mu - \sigma^2/2)dt + \sigma dW_t$. We can obtain the following discretized version and use it to generate stock price sample paths: $S_{t+1} = S_t \exp\{(\mu - \sigma^2/2)\Delta t + \sigma\sqrt{\Delta t}\varepsilon\}$, where Δt is a small time step, and $\varepsilon \sim N(0,1)$, the standard normal distribution. To estimate the constant σ from real data, we use the method of maximum likelihood estimation (MLE).

GARCH model. In a GBM, the volatility is assumed to be a constant. In reality, the volatility may itself be stochastic. We use GARCH(1,1) as a stochastic volatility model: $\sigma_t^2 = \omega + \alpha u_{t-1}^2 + \beta\sigma_{t-1}^2$, where $u_t = \ln(S_t/S_{t-1})$, and α and β are weights for u_{t-1}^2 and σ_{t-1}^2 respectively. It is required that $\alpha + \beta < 1$ for the stability of GARCH(1,1). The constant ω is related to the long term average volatility σ_L by $\omega = (1 - \alpha - \beta)\sigma_L$. The discretized version is: $S_{t+1} = S_t \exp\{(\mu - \sigma_t^2/2)\Delta t + \sigma_t\sqrt{\Delta t}\varepsilon\}$. To estimate the parameters for a GARCH model and to generate sample paths, we use the MATLAB GARCH toolbox functions 'garchfit' and 'garchsim'.

6.2 Basis functions

LSPI, FQI and LSM need to choose basis functions to approximate the expected continuation value. As suggested in Longstaff and Schwartz (2001),we use $\varphi_0(S) = 1$ and the following Laguerre polynomials to generalize over the stock price: $\varphi_1(S) = \exp(-S'/2)$, $\varphi_2(S) = \exp(-S'/2)(1-S')$, and $\varphi_3(S) = \exp(-S'/2)(1-2S'+S'^2/2)$. We use $S' = S/K$ instead of S in the basis functions, where K is the strike price,

since the function $\exp(-S/2)$ goes to zero fast. LSPI and FQI also generalize over time t. We use the following functions for time t: $\varphi_0^t(t) = \sin(-t\pi/2T + \pi/2)$, $\varphi_1^t(t) = \ln(T-t)$, $\varphi_2^t(t) = (t/T)^2$, guided by the observation that the optimal exercise boundary for an American put option is a monotonic increasing function, as shown in Duffie (2001).

American put options. As we already noted, the intrinsic value of an American stock put option is $g(S) = \max(0, K-S)$. LSM uses the functions $\varphi_0(S)$, $\varphi_1(S)$, $\varphi_2(S)$, and $\varphi_3(S)$. LSM computes different sets of weight vectors for the basis functions for different time steps. LSPI and FQI use the functions: $\varphi_0(S,t) = \varphi_0(S)$, $\varphi_1(S,t) = \varphi_1(S)$, $\varphi_2(S,t) = \varphi_2(S)$, $\varphi_3(S,t) = \varphi_3(S)$, $\varphi_4(S,t) = \varphi_0^t(t)$, $\varphi_5(S,t) = \varphi_1^t(t)$, and $\varphi_6(S,t) = \varphi_2^t(t)$. LSPI (FQI) determines a single weight vector over all time steps to calculate the continuation value.

LSM only uses the 4 features over the prices (giving altogether $4(T-2)$ weights), while LSPI and FQI uses both sets of basis functions (i.e., 7 weights). Note that neither LSPI, nor FQI models correlations directly between time and stock prices, i.e., the decomposition of the continuation values corresponds to the first level of an ANOVA decomposition.

6.3 Results for American put options

For real data, a pricing method can learn an exercise policy either 1) from sample paths generated from a simulation model; or, 2) from sample paths composed from real data directly. The testing sample paths are from real data and separate from data used in trainings. We scale the stock prices, so that, for each company, the initial price for each training path and each testing path is the same as the first price of the whole price series of the company.

Now we proceed with the first approach. The simulation model for the underlying stock process follows a GBM model or a GARCH model, with parameters estimated from real data. For options with quarterly, semi-annual and annual maturities respectively, the first 662, 625 and 751 stock prices are used for estimating parameters. Then LSPI, FQI and LSM learn exercise policies with (the same) 50,000 sample paths. We call this approach of generating sample paths from a simulation model with parameters estimated from real data as LSPI_gbm, LSPI_garch, FQI_gbm, FQI_garch, LSM_gbm and LSM_garch, respectively.

In the second approach, a pricing method learns the exercise policy from sample paths composed from real data directly. Due to the scarcity of real data, as there is only a single trajectory of stock price time series for each company, we construct multiple trajectories following a windowing technique. For each company, for quarterly, semi-annual, annual maturity terms, we obtain 600, 500, 500 training paths, each with *duration* = 63, 126, 252 prices. The first path is the first *duration* days of stock prices. Then we move one day ahead and obtain the second path, and so on. LSPI and LSM then learn exercise policies on these training paths. We call this approach of generating sample paths from real data directly as LSPI_data, FQI_data and LSM_data, respectively.

After the exercise policies are found, we compare their performances on the test paths. For each company, for the quarterly, semi-annual and annual maturity terms, we obtain 500, 450, 250 testing paths, each with *duration* = 63, 126, 252 prices, as follows. The first path is the last *duration* days of stock prices. Then we move one day back and obtain the second path, and so on.

For each maturity term of each of the Dow Jones 30 companies, we average payoffs over the testing paths. Then we average the average payoffs over the 30 companies. Table 1 presents the average results. These results show that LSPI and FQI gain larger average payoffs than LSM. These results are (highly) significant when test by a paired two-tailed t-test. The differences between the performances of LSPI and FQI are not significant.

A likely explanation for why LSPI and FQI are gaining larger payoffs than LSM is that LSPI and FQI generalize across all time steps and use a parsimonious representation, whereas LSM uses many weights and does not attempt to generalize across time. Since the training data is typically limited, the parsimonious representation can reduce estimation errors, even though the approximation error of this representation might be larger (see Theorem 1 for the tradeoff between the two terms).

The results in Table 1 also show that LSPI_data slightly outperforms both LSPI_gbm and LSPI_garch. The same holds for FQI. However, a paired t-test indicates that the observed differences are not significant (in the case of annual maturity term the difference in the results obtained on the MLE data and the real data is "marginal", but still not significant). The differences in the performance obtained with LSM when it is trained with the real data and when it is trained with models is closer to be significant in all cases ($p \approx 0.25$). Hence, in this case it seems that the models do not introduce significant biases, or these biases are compensated by the larger number of trajectories available for training the exercise policies.

In Figure 1, we present the exercise boundaries discovered by LSPI, FQI and LSM for the real data of

maturity	LSPI			FQI			LSM		
	gbm	garch	data	gbm	garch	data	gbm	garch	data
quarterly	1.310	1.333	1.339	1.321	1.341	1.331	0.573	0.572	0.719
semi-annual	1.681	1.663	1.739	1.718	1.749	1.797	0.693	0.687	0.887
annual	1.599	1.496	1.677	1.832	1.797	2.015	0.717	0.685	0.860

Table 1: Average payoffs of LSPI, FQI and LSM on real data for Dow Jones 30 companies.

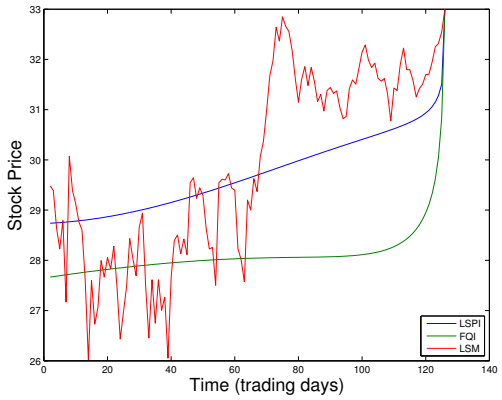

Figure 1: Exercise boundaries.

Intel, with semi-annual maturity and $r = 0.03$. The optimal exercise boundary for an American put option is a monotonic increasing function, as shown in Duffie (2001). Figure 1 shows that unlike the boundary discovered by LSM, the exercise boundaries discovered by LSPI and FQI are smooth and (mostly) respect monotonicity. The scarcity of sample paths may explain this non-monotonicity. The boundary of FQI is lower than that of LSPI, which is typical according to our experience and we are currently investigating the cause of this. With 50,000 training paths (with simulated data), the exercise boundaries discovered by LSM will be much smoother, but still choppy.

We also evaluate LSPI, FQI and LSM with synthetic sample paths, with the parameters for the GBM model and the GARCH model estimated from real data. For each company, we generate 50,000 training paths 10,000 testing paths. The results in Table 2 indicate that LSPI and FQI gain larger payoffs than LSM, both in the GBM model and in the GARCH model (the interest rate is $r = 0.03$). Also, in the GBM model the difference between LSM and the two methods is smaller.

Besides American put stock options, we also study American Asian stock options, which are complex, exotic, path-dependent options, whose payoffs are determined by the average of past stock prices. The results are similar.

7 Conclusions

Options are important financial instruments and pricing of complex options is a non-trivial problem. Our empirical results show that methods developed in RL can advance the state-of-the-art in this challenging application area and demonstrates that computational finance remains a promising area for future applications of RL methods. However, much remains to be done. The theoretical analysis presented leaves open the question of how to choose the function approximation technique. When data is scarce, controlling both the estimation and the approximation error becomes important. This calls for model-selection technique. A simple solution is to generate policies with different function approximators and compare their performance. Unlike in a general MDP this is possible because given the price trajectories one can simulate any policy on *real data*. Hence, separating some test data should make it possible to do model-selection in an efficient way. However, this is left as future work.

Appendix

Here we sketch the proof for Theorem 1, which refines the proof of Theorem 4 in Antos et al. (2008) for the LSPI option pricing algorithm.

First note that in the light of Proposition 2 of Antos et al. (2008), this theorem can indeed be applied to LSPI. The major modification is that due to the finite horizon nature of the problem all the work has to be done with sequences of sub-probability measures (of length T) instead of working with probability measures over $\mathcal{S}$. Accordingly, the stochastic kernels, P_π, associated with policies must be redefined. In particular, P_π, as a left-linear operator will assign the sub-probability measure sequence $(\mu'_0, \dots, \mu'_{T-1})$ to a sub-probability measure sequence $(\mu_0, \dots, \mu_{T-1})$ as follows: Here measures μ_t, μ'_t are over $\mathcal{S}_t \times \mathcal{A}$. First, let $\tilde{\mu}_t = \mu_t / \mu_t(\mathcal{S}_t \times \mathcal{A})$. Let $U \subset \mathcal{S}_{t+1} \times \mathcal{A}$ be an arbitrary measurable set. Then $\mu'_{t+1}(U) = \mathbb{P}((S''_t, A''_t) \in U)$. Here the random variables S''_t, A''_t are defined as follows: Let $(S_t, A_t) \sim \tilde{\mu}_t(\cdot)$, $S'_t \sim \mathcal{P}_t(\cdot|S_t)$, $S''_t = S'_t$ iff $A_t = 0$, $S''_t = e$, otherwise, and $A''_t = \pi(S''_t)$. Further, $\mu'_0(U) = \mathbb{P}((S_0, A_0) \in U)$, where $S_0 \sim \mathcal{P}_0(\cdot)$, $A_0 = \pi(S_0)$. The kernel P_π is also interpreted as a

maturity term	GBM model			GARCH model		
	LSPI	FQI	LSM	LSPI	FQI	LSM
quarterly	2.071	2.054	2.044	1.889	1.866	0.785
semi-annual	2.771	2.758	2.742	2.546	2.530	0.997
annual	3.615	3.645	3.580	3.286	3.311	1.241

Table 2: Average payoffs on synthetic data with parameters estimated from real data.

right linear operator acting on value functions. For this, first define the "integral" of $Q \in B(\mathcal{S} \times \mathcal{A})$ with respect to μ, where $\mu = (\mu_0, \ldots, \mu_{T-1})$ as above. Then $\mu Q = (\mu_0 Q_0, \ldots, \mu_{T-1} Q_{T-1})$, where, as usual, for a measure μ_t over $\mathcal{S}_t \times \mathcal{A}_t$ and a function Q_t over the same domain $\mu_t Q_t = \int Q_t(s,a)\mu_t(ds, da)$. Then $Q' = P_\pi Q \in B(\mathcal{S} \times \mathcal{A})$ is defined as follows: $Q'(e, a) = 0$, $Q'((s,t), a) = \delta_{s,a} Q_t$, $s \in \mathcal{S}_t$, $t = 0, \ldots, T-1$. These definitions allow us to repeat the proof of Theorem 4 of Antos et al. (2008) (in particular, it makes Lemma 12 of Antos et al. (2008) hold). One more change that is required is to redefine the m-step future-state concentrability coefficients $c_{\rho,\nu}(m)$ (needed in the proof of Lemma 12 and influencing the bound of Theorem 4 of Antos et al. (2008)). The modified definition is $c_{\rho,\nu}(m) = \sup_{\pi_1,\ldots,\pi_m} d/d\nu(\rho P_{\pi_1} \ldots P_{\pi_m})_{\mathcal{S}_c}$, where ρ is a distribution sequence over the state-action pairs, ν is a distribution sequence over $\mathcal{S}$, and $\mathcal{S}_c = \{ (s, 0) : s \in \mathcal{S} \} \subset \mathcal{S} \times \mathcal{A}$, which can be (and is) identified with the set $\mathcal{S}$.

The main observation is the following: Let π be a policy and let $\hat{\mu}_\pi$ be such that $\hat{\mu}_{\pi,t}(U) = \mathbb{P}((S_t, A_t) \in U)$, where S_t is a price process obtained with the kernels $(\mathcal{P}_t)_{t \geq 0}$ and $A_t = \pi(\cdot|S_t)$. Then for *any* policies π, π', as it follows directly from the definitions, $\hat{\mu}_\pi P_{\pi'} \leq \hat{\mu}_{\pi'}$, where for the sequences the comparison happens componentwise and for measures ν, ρ defined over a common domain, $\nu \leq \rho$ if $\nu(U) \leq \rho(U)$ holds for all measurable set U in the common domain. Hence, if $\nu = \rho = \hat{\mu}_{\pi_c}$, where π_c always selects action 0, $\rho P_{\pi_1} = \hat{\mu}_{\pi_c} P_{\pi_1} \leq \hat{\mu}_{\pi_1}$, thus $\rho P_{\pi_1} P_{\pi_2} \leq \hat{\mu}_{\pi_1} P_{\pi_2} \leq \hat{\mu}_{\pi_2}$. Continuing this way we get $\rho P_{\pi_1} \ldots P_{\pi_m} \leq \hat{\mu}_{\pi_m}$. Hence, $(\rho P_{\pi_1} \ldots P_{\pi_m} \leq \hat{\mu}_{\pi_m})|_{\mathcal{S}_c} \leq \hat{\mu}_{\pi_c}$. Hence, with this choice, $c_{\rho,\nu}(m) \leq 1$. This gives $C_{\rho,\nu} = (1-\gamma)^2 \sum_{m \geq 1} m \gamma^{m-1} c_{\rho,\mu}(m) \leq 1$. Further, one can show that the β-mixing condition of Theorem 4 is satisfied by choosing $\kappa = 1$, $b = 1$ and $\overline{\beta}$ such that $\log(\overline{\beta}) = T$ (this is the origin of T in Λ). The pseudo-dimension of the function spaces and their VC-crossing dimension (by Proposition 3 of Antos et al. (2008)) are equal to d, again appearing in Λ.

References

Andras Antos, Csaba Szepesvari, and Remi Munos. Learning near-optimal policies with bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning Journal*, 71:89–129, 2008.

Dimitri P. Bertsekas. *Dynamic programming and optimal control*. Athena Scientific, Massachusetts, USA, 1995.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Massachusetts, USA, 1996.

Steven J. Bradtke and Andrew G. Barto. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22(1-3):33–57, March 1996.

Mark Broadie and Jerome B. Detemple. Option pricing: valuation models and applications. *Management Science*, 50(9):1145–1177, September 2004.

Darrell Duffie. *Dynamic asset pricing theory*. Princeton University Press, 2001.

Paul Glasserman. *Monte Carlo Methods in Financial Engineering*. Springer-Verlag, New York, 2004.

John C. Hull. *Options, Futures and Other Derivatives (6th edition)*. Prentice Hall, 2006.

Michail G. Lagoudakis and Ronald Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107 – 1149, December 2003.

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

R. Munos and Cs. Szepesvári. Finite time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9:815–857, 2008.

Martin L. Puterman. *Markov decision processes : discrete stochastic dynamic programming*. John Wiley & Sons, New York, 1994.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

John. N. Tsitsiklis and Benjamin Van Roy. Regression methods for pricing complex American-style options. *IEEE Transactions on Neural Networks (special issue on computational finance)*, 12(4):694–703, July 2001.