Policy Iteration for Learning an Exercise Policy for American Options

Yuxi Li, Dale Schuurmans

Department of Computing Science, University of Alberta

Abstract. Options are important financial instruments, whose prices are usually determined by computational methods. Computational finance is a compelling application area for reinforcement learning research, where hard sequential decision making problems abound and have great practical significance. In this paper, we investigate reinforcement learning methods, in particular, least squares policy iteration (LSPI), for the problem of learning an exercise policy for American options. We also investigate TVR, another policy iteration method. We compare LSPI, TVR with LSM, the standard least squares Monte Carlo method from the finance community. We evaluate their performance on both real and synthetic data. The results show that the exercise policies discovered by LSPI and TVR gain larger payoffs than those discovered by LSM, on both real and synthetic data. Furthermore, for LSPI, TVR and LSM, policies learned from real data generally gain larger payoffs than policies learned from simulated samples. Our work shows that solution methods developed in reinforcement learning can advance the state of the art in an important and challenging application area, and demonstrates furthermore that computational finance remains an under-explored area for deployment of reinforcement learning methods.

1 Introduction

Options are an essential financial instrument for hedging and risk management, and therefore, options pricing and finding optimal exercise policies are important problems in finance.[1] Options pricing is usually approached by computational methods. In general, computational finance is a compelling application area for reinforcement learning research, where hard sequential decision making problems abound and have great practical significance [11]. In this paper, we show solution techniques from the reinforcement learning literature are superior to a standard technique from the finance literature for pricing American options, a classical sequential decision making problem in finance.

Options pricing is an optimal control problem, usually modeled as Markov Decision Processes (MDP). Dynamic programming is a method to find an optimal policy for an MDP [2, 12], usually with the model of the MDP. When the

[1] A call/put option gives the holder the right, not the obligation, to buy/sell the underlying asset, for example, a share of a stock, by a certain date (maturity date) for a certain price (strike price). An American option can be exercised any time up to the maturity date.

size of an MDP is large, for example, when the state space is continuous, we encounter the "curse of dimensionality". Reinforcement learning, also known as neuro-dynamic programming, is an approach to addressing this scaling problem, and can work without a model of the MDP [3, 13]. Successful investigations include the application of reinforcement learning to playing backgammon, dynamic channel allocation, elevator dispatching, and so on. The key idea behind these successes is to exploit effective approximation methods. Linear approximation has been most widely used. A reinforcement learning method can learn an optimal policy for an MDP either from simulated samples or directly from real data. One advantage of basing directly an approximation architecture on the underlying MDP is that the error for the simulation model is eliminated.

In the community of computational finance, researchers have investigated pricing methods using analytic models and numerical methods, including the risk-neutral approach, the lattice and finite difference methods, and the Monte Carlo methods. For example, Hull [8] provides an introduction to options and other financial derivatives and their pricing methods, Broadie and Detemple [5] survey option pricing methods, and Glasserman [7] provides a book length treatment for Monte Carlo methods. Most of these methods follow the backward-recursive approach of dynamic programming. Two examples that deploy approximate dynamic programming for the problem of pricing American options are: the least squares Monte Carlo (LSM) method in [10] and the approximate value iteration approach in [14].

Our goal is to investigate reinforcement learning type algorithms for pricing American options. In this work, we extend an approximate policy iteration method, namely, least squares policy iteration (LSPI) in [9], to the problem of pricing American options. We also investigate the policy iteration method proposed in [14], referred to as TVR. We empirically evaluate the performance of LSPI, TVR and LSM, with respect to the payoffs the exercise policies gain. In contrast, previous work evaluates pricing methods by measuring the accuracy of the estimated prices. The results show that, on both real and synthetic data, exercise policies discovered by LSPI and TVR can achieve larger payoffs than those found by LSM. Furthermore, for LSPI, TVR and LSM, policies discovered based on sample paths composed directly from real data gain larger payoffs than the policies discovered based on sample paths generated by simulation models with model parameters estimated from real data.

In this work, we present a successful application of reinforcement learning research, the policy iteration method, for learning an exercise policy for American options, and show its superiority to LSM, the standard option pricing method in finance. As well, we introduce a new performance measure, the payoff a pricing method gains, for comparing option pricing methods in the empirical study.

The reminder of this paper is organized as follows. First, we introduce MDPs and LSPI. Then, we present the extension of LSPI to pricing American options, and introduce TVR and LSM. After that, we study empirically the performance of LSPI, TVR and LSM on both real and synthetic data. Finally, we conclude.

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. An MDP is defined by the 5-tuple (S, A, P, R, γ). S is a set of states; A is a set of actions; P is a transition model, with $P(s, a, s')$ specifying the conditional probability of transitioning to state s' starting from state s and taking action a; R is a reward function, with $R(s, a, s')$ being the reward for transition to state s' starting from state s and taking action a; and γ is a discount factor.

A policy π is a rule for selecting actions based on observed states. $\pi(s, a)$ specifies the probability of selecting action a in state s by following policy π. An optimal policy maximizes the rewards obtained over the long run. We define the long run reward in an MDP as maximizing the infinite horizon discounted reward $\sum_{t=0}^{\infty} \gamma^t r_t$ obtained over an infinite run of the MDP, given a discount factor $0 < \gamma < 1$. A policy π is associated with a value function for each state-action pair (s, a), $Q^\pi(s, a)$, which represents the expected, discounted, total reward starting from state s taking action a and following policy π thereafter. That is, $Q^\pi(s, a) = E(\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_0 = a)$, where the expectation is taken with respect to policy π and the transition model P. Q^π can be found by solving the following linear system of Bellman equations: $Q^\pi(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s, a, s') \sum_{a' \in A} \pi(s', a') Q^\pi(s', a')$, where $R(s, a) = \sum_{s'} P(s, a, s') R(s, a, s')$ is the expected reward for state-action pair (s, a). Q^π is the fixed point of the Bellman operator T_π: $(T_\pi Q)(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s, a, s') \sum_{a' \in A} \pi(s', a') Q(s', a')$. T_π is a monotonic operator and a contraction mapping in the L_∞-norm. The implication is that successive application of T_π for any initial Q converges to Q^π. This is value iteration, a principle method for computing Q^π.

When the size of an MDP becomes large, its solution methods encounter the "curse of dimensionality". Approximation architecture is an approach to addressing the scalability concern. The linear architecture is an efficient and effective approach. In the linear architecture, the approximate value function is represented by:[2] $\hat{Q}^\pi(s, a; w) = \sum_{i=1}^{k} \phi_i(s, a) w_i$, where $\phi_i(\cdot, \cdot)$ is a basis function, w_i is its weight, and k is the number of basis functions. Define

$$\phi(s, a) = \begin{pmatrix} \phi_1(s, a) \\ \phi_2(s, a) \\ \cdots \\ \phi_k(s, a) \end{pmatrix}, \mathbf{\Phi} = \begin{pmatrix} \phi(s_1, a_1)^\mathsf{T} \\ \cdots \\ \phi(s, a)^\mathsf{T} \\ \cdots \\ \phi(s_{|S|}, a_{|A|})^\mathsf{T} \end{pmatrix}, w^\pi = \begin{pmatrix} w_1^\pi \\ w_2^\pi \\ \cdots \\ w_k^\pi \end{pmatrix},$$

where T denotes matrix transpose. $\hat{Q}^\pi$ then can be represented as $\hat{Q}^\pi = \mathbf{\Phi} w^\pi$.

Least squares policy iteration. Policy iteration is a method of discovering an optimal solution for an MDP. LSPI [9] combines the data efficiency of the least squares temporal difference method [4] and the policy search efficiency of

[2] Following the conventional notation, an approximate representation is denoted with the $\hat{}$ symbol, and a learned estimate is denoted with the $\tilde{}$ symbol.

policy iteration. Next, we give a brief introduction of LSPI.[3] The matrix form of the Bellman equation is: $Q^\pi = R + \gamma \mathbf{P}\mathbf{\Pi}_\pi Q^\pi$, where $\mathbf{P}$ is a $|S||A| \times |A|$ matrix, with $\mathbf{P}((s,a),s') = P(s,a,s')$, and $\mathbf{\Pi}$ is a $|S| \times |S||A|$ matrix, with $\mathbf{\Pi}(s',(s',a')) = \pi(s',a')$.

The state-action value function Q^π is the fixed point of the Bellman operator: $T_\pi Q^\pi = Q^\pi$. An approach to finding a good approximation is to force $\hat{Q}^\pi$ to be a fixed point of the Bellman operator: $T_\pi \hat{Q}^\pi \approx \hat{Q}^\pi$. $\hat{Q}^\pi$ is in the space spanned by the basis functions. However, $T_\pi \hat{Q}^\pi$ may not be in this space. LSPI requires that, $\hat{Q}^\pi = \mathbf{\Phi}(\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}(T_\pi \hat{Q}^\pi) = \mathbf{\Phi}(\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}(R + \gamma \mathbf{P}\mathbf{\Pi}_\pi Q^\pi)$, where, $\mathbf{\Phi}(\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}$ is the orthogonal projection which minimizes the L_2-norm. From this, we can obtain, $w^\pi = \left(\mathbf{\Phi}^\mathsf{T}(\mathbf{\Phi} - \gamma \mathbf{P}\mathbf{\Pi}_\pi \mathbf{\Phi})\right)^{-1} \mathbf{\Phi}^\mathsf{T} R$. The weighted least squares fixed point solution is: $w^\pi = \left(\mathbf{\Phi}^\mathsf{T}\Delta_\mu(\mathbf{\Phi} - \gamma \mathbf{P}\mathbf{\Pi}_\pi \mathbf{\Phi})\right)^{-1} \mathbf{\Phi}^\mathsf{T}\Delta_\mu R$, where Δ_μ is the diagonal matrix with the entries of $\mu(s,a)$, which is a probability distribution over state-action pairs $(S \times A)$. This can be written as $\mathbf{A}w^\pi = b$, where $\mathbf{A} = \mathbf{\Phi}^\mathsf{T}\Delta_\mu(\mathbf{\Phi} - \gamma \mathbf{P}\mathbf{\Pi}_\pi \mathbf{\Phi})$ and $b = \mathbf{\Phi}^\mathsf{T}\Delta_\mu R$.

Without a model of the MDP, that is, without the full knowledge of $\mathbf{P}$, $\mathbf{\Pi}_\pi$ and R, we need a learning method to discover an optimal policy. It is shown in [9] that $\mathbf{A}$ and b can be learned incrementally as, at iteration $t+1$:

$$\tilde{\mathbf{A}}^{(t+1)} = \tilde{\mathbf{A}}^{(t)} + \phi(s_t)(\phi(s_t) - \gamma\phi(s_{t+1}))^\mathsf{T} \text{ and } \tilde{b}^{(t+1)} = \tilde{b}^{(t)} + \phi(s_t)R_t \qquad (1)$$

The boundedness property of LSPI is established in [9] with respect to the L_∞-norm. Recently, a tighter bound is given in [1] for policy iteration with continuous state spaces on a single sample path.

3 Learning an exercise policy for American options

We first discuss the application of LSPI for the problem of learning an exercise policy for American options. Next we give a brief review of TVR [14] and LSM [10]. We discretize the time, thus the options become Bermudan.

3.1 LSPI for learning an exercise policy for American options

We need to consider several peculiarities of the problem of learning an exercise policy for American options, when applying LSPI for it. First, it is an episodic, optimal stopping problem. It may terminate any time between the starting date and the maturity date of the option. Usually, after a termination decision is made, LSPI needs to start over from a new sample path. This is data inefficient. We use the whole sample path, even in the case the option is exercised at an intermediate time step following the current policy. Second, in option pricing, the continuation value of an option may be different at different time, even with the same underlying asset price and other factors. Thus we incorporate

[3] This is the LPSI with least-squares fixed-point approximation. LSPI can also work with Bellman residual minimizing approximation, which we do not discuss here.

time as a component in the state space. Third, there are two actions for each state, exercise and continue. The state-action value function of exercising the option, that is, the intrinsic value of the option, can be calculated exactly. We only need to consider the state-action value function for continuation, that is, $Q(s, a = \text{continue})$. Fourth, before exercising an option, there is no reward to the option holder, that is, $R = 0$. When the option is exercised, the reward is the payoff.

3.2 TVR: the policy iteration approach in [14]

We introduce TVR [14] in the following. We use $Q(S,t)$ to denote $Q(\{S,t\}, a = \textit{continue})$, where S is the stock price. We want to find a projection Π of $Q = (Q(S,0), Q(S,1), \ldots, Q(S,T-1))$ in the form $\mathbf{\Phi}w$, where w is to minimize $\sum_{t=0}^{T-1} E[(\mathbf{\Phi}(S_t,t)w - Q(S_t,t))^2]$, where the expectation $E[(\mathbf{\Phi}(S_t,t)w - Q(S_t,t))^2]$ is with respect to the probability measure of S_t. The weight w is given by

$$w = \left(\sum_{t=0}^{T-1} E[\mathbf{\Phi}(S_t,t)\mathbf{\Phi}^{\mathsf{T}}(S_t,t)] \right)^{-1} E[\mathbf{\Phi}(S_t,t)Q(S_t,t)] \tag{2}$$

Define $g(S)$ as the intrinsic value of the option when the stock price is S, and $J_t(S)$ as the price of the option at time t when $S_t = S$: $J_T = g$ and $J_t = \max(g, \gamma \mathcal{P} J_{t+1}), t = T-1, T-2, \ldots, 0$, where $(\mathcal{P}J)(S) = E[J(S_{t+1})|S_t = S]$. Define $FJ = \gamma \mathcal{P} \max(g, J)$. We have $(Q(\cdot,0), Q(\cdot,1), \ldots, Q(\cdot,T-1)) = (FQ(\cdot,1), FQ(\cdot,2), \ldots, FQ(\cdot,T))$, which is denoted compactly as $Q = HQ$. The above solution of w is thus the fixed point of the equation $HQ^* = Q^*$. It is difficult to solve this function, since Q^* is unknown. We resort to the fixed point of equation $Q = \Pi HQ$. Suppose w_i is the weight vector computed at iteration i (w_0 can be arbitrarily initialized),

$$w_{i+1} = \left(\sum_{t=0}^{T-1} E[\mathbf{\Phi}(S_t,t)\mathbf{\Phi}^{\mathsf{T}}(S_t,t)] \right)^{-1} E[\mathbf{\Phi}(S_t,t) \max(g(S_{t+1}), \mathbf{\Phi}(S_{t+1}^j,t)w_i)] \tag{3}$$

The expectation with respect to the underlying probability measure can be replaced with an expectation with respect to the empirical measure provided by unbiased samples. The following is an implementable version with sample trajectories $S_t^j, j = 1, \ldots, m$, where S_t^j is the value of S_t in the j-th trajectory:

$$\hat{w}_{i+1} = \left(\sum_{t=0}^{T-1} \sum_{j=1}^{m} \mathbf{\Phi}(S_t^j,t)\mathbf{\Phi}^{\mathsf{T}}(S_t^j,t) \right)^{-1} \sum_{t=0}^{T-1} \sum_{j=1}^{m} \mathbf{\Phi}(S_t^j,t) \max(g(S_{t+1}^j), \mathbf{\Phi}(S_{t+1}^j,t)\hat{w}_i) \tag{4}$$

3.3 Least squares Monte Carlo

LSM in [10] follows the backward-recursive dynamic programming approach with function approximation of expected continuation value. It estimates the expected

continuation value from the second-to-last time step backward until the first time step, on the sample paths. At each time step, LSM fits the expected continuation value on the set of basis functions with least squares regression, using the cross-sectional information from the sample paths and the previous iterations (or the last time step). Specifically, at time step t, assuming the option is not exercised, the continuation values for the sample paths (LSM uses only in-the-money paths) can be computed, since in a backward-recursive approach, LSM has already considered time steps after t until the maturity. As well, values of the basis functions can be evaluated for the asset prices at time step t. Then, LSM regresses the continuation values on the values of the basis functions with least squares, to obtain the weights for the basis functions for time step t. When LSM reaches the first time step, it obtains the price of the option. LSM also obtains the weights for the basis functions for each time step. These weights represent implicitly the exercising policy. The approximate value iteration method in [14] is conceptually similar to LSM. (TVR is also proposed in [14].)

4 Empirical study

We study empirically the performance of LSPI, TVR and LSM on learning an exercise policy for American options. We study the plain vanilla American put stock options and American Asian 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. In LSPI, we set the discount factor $\gamma = e^{-r/252}$. LSPI and TVR iterate on the sample paths until the difference between two successive policies is sufficiently small, or when it has run 15 iterations (LSPI and TVR 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. We study the payoff a policy gain, which is the intrinsic value of an option when the option is exercised.

4.1 Simulation models

In our experiments when a simulation model is used, synthetic data may be generated from either the geometric Brownian Motion (GBM) model or a stochastic volatility (SV) model, two of the most widely used models for stock price movement. See [8] for detail.

Geometric Brownian motion model. Suppose S_t, the stock price at time t, follows a GBM:

$$dS_t = \mu S_t dt + \sigma S_t dW_t, \tag{5}$$

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:

$$d\ln S_t = (\mu - \sigma^2/2)dt + \sigma dW_t. \tag{6}$$

We can obtain the following discretized version for (6), and use it to generate stock price sample paths:

$$S_{t+1} = S_t \exp\{(\mu - \sigma^2/2)\Delta t + \sigma\sqrt{\Delta t}\epsilon\}, \tag{7}$$

where Δt is a small time step, and $\epsilon \sim N(0,1)$, the standard normal distribution.

To estimate the constant σ from real data, we use the method of maximum likelihood estimation (MLE).

Stochastic volatility model. In the 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:

$$\sigma_t^2 = \omega + \alpha u_{t-1}^2 + \beta\sigma_{t-1}^2, \tag{8}$$

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:

$$S_{t+1} = S_t \exp\{(\mu - \sigma_t^2/2)\Delta t + \sigma_t\sqrt{\Delta t}\epsilon\}. \tag{9}$$

To estimate the parameters for the SV model in (8) and to generate sample paths, we use the MATLAB GARCH toolbox functions 'garchfit' and 'garchsim'.

4.2 Basis functions

LSPI, TVR and LSM need to choose basis functions to approximate the expected continuation value. As suggested in [10],we use the constant $\phi_0(S) = 1$ and the following Laguerre polynomials to generalize over the stock price: $\phi_1(S) = \exp(-S'/2)$, $\phi_2(S) = \exp(-S'/2)(1 - S')$, and $\phi_3(S) = \exp(-S'/2)(1 - 2S' + S'^2/2)$. 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 TVR also generalize over time t. We use the following functions for time t: $\phi_0^t(t) = \sin(-t\pi/2T + \pi/2)$, $\phi_1^t(t) = \ln(T - t)$, $\phi_2^t(t) = (t/T)^2$, guided by the observation that the optimal exercise boundary for an American put option is a monotonic increasing function, as shown in [6].

American stock put options. The intrinsic value of an American stock put options is $g(S) = \max(0, K - S)$. LSM uses the functions $\phi_0(S)$, $\phi_1(S)$, $\phi_2(S)$, and $\phi_3(S)$. LSM computes different sets of weight vectors for the basis functions for different time steps. LSPI and TVR use the functions: $\phi_0(S,t) = \phi_0(S)$, $\phi_1(S,t) = \phi_1(S)$, $\phi_2(S,t) = \phi_2(S)$, $\phi_3(S,t) = \phi_3(S)$, $\phi_4(S,t) = \phi_0^t(t)$, $\phi_5(S,t) = \phi_1^t(t)$, and $\phi_6(S,t) = \phi_2^t(t)$. LSPI (TVR) determines a single weight vector over all time steps to calculate the continuation value.

American Asian call options. Asian options are exotic, path-dependent options. We consider a call option whose payoff is determined by the average price Avg of a stock over some time horizon, and the option can be exercised at any time after some initial lockout time period. The intrinsic value is $g(Avg) = \max(0, Avg - K)$. The choice of the eight basis functions for a stock price and the average of stock price follows the suggestion in [10]: a constant, the first two Laguerre polynomials for the stock price, the first two Laguerre polynomials for the average stock price, and the cross products of these Laguerre polynomials up to third order terms. LSPI and TVR take time as a component in the state space. We use the same set of basis functions for time t as those used for the American stock put options.

4.3 Results for American put options: real data

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. 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 the GBM in (5) or the SV model in (8). For the GBM model, the constant volatility σ is estimated from the training data with MLE. For the SV model, we use the popular GARCH(1,1) to estimate the parameters, ω, α and β in (8). In this case, for options with quarterly, semi-annual and annual maturities respectively, the first 662, 625 and 751 stock prices are used for estimating parameters in (5) and in (8). Then LSPI, TVR and LSM learn exercise policies with 50,000 sample paths, generated using the models in (5) or in (8) with the estimated parameters. We call this approach of generating sample paths from a simulation model with parameters estimated from real data as LSPI_mle, LSPI_garch, TVR_mle, TVR_garch, LSM_mle 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, TVR_data and LSM_data, respectively.

After the exercise policies are found by LSPI, TVR and LSM, we compare their performance on testing paths. For each company, for quarterly, semi-annual, 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 shows the results for each company and the average over 30 companies for semi-annual maturity. Table 2 presents the average results. These results show that LSPI and TVR gain larger average payoffs than LSM.

An explanation for LSPI and TVR gaining larger payoffs is, LSPI and TVR optimize weights across all time steps, whereas LSM is a value iteration procedure that makes a single backward pass through time. Thus, LSPI and TVR are able to eliminate some of the local errors. With the same sample paths, LSPI and TVR have the chance to improve a policy in an iterative approach. Thus, the policy learned by LSPI and TVR will ultimately converge to an optimal policy supported by the basis functions. However, LSM works in the backward-recursive approach. After LSM determines a policy with the least squares regression, it does not improve it.

LSM computes different sets of weights for the basis functions for different time steps; thus it generalizes over the space for asset prices. In contrast, LSPI and TVR deploy function approximation for both stock price and time, so that they generalize over both the space for asset prices and the space for time. Therefore LSM has a stronger representation than LSPI and TVR. However, LSPI and TVR outperform LSM.

The results in Table 1 and Table 2 also show that LSPI_data outperforms both LSPI_mle and LSPI_garch. That is, in the studied cases, an exercise policy learned by LSPI with sample paths composed directly from real data gains larger payoffs on average than an exercise policy learned by LSPI with sample paths generated from either the GBM model in (5) or the SV model in (8), with model parameters estimated from real data. Note, the set of real data to generate sample paths for LSPI_data is the same as the set of real data to estimated parameters for either the GBM model or the SV model. As well, the results also show that LSM_data outperforms both LSM_mle and LSM_garch. For TVR, except the quarterly case, TVR_data outperforms both TVR_mle and TVR_garch.

We believe the key reason that LSPI_data outperforms LSPI_mle and LSPI_garch is that LSPI_data learns the exercise policy from real data directly, without estimating parameters for a simulation model first. In this way, LSPI_data eliminates the errors in estimating the model parameters, as encountered by LSPI_mle and LSPI_garch. This explanation applies similarly to the results for LSM and TVR.

4.4 Results for American put options: synthetic data

We evaluate the performance of LSPI, TVR and LSM with synthetic sample paths. The parameters for the GBM model in (5) and the SV model in (8) can either 1) be estimated from real data; or, 2) be set in some arbitrary manner. The training sample paths and the testing sample paths are generated using the same model with the same parameters.

Name	LSPI			TVR			LSM		
	mle	garch	data	mle	garch	data	mle	garch	data
3M	2.448	2.404	3.329	2.852	2.370	3.477	0.944	0.944	1.194
Alcoa	2.403	2.400	2.361	2.414	2.403	2.362	0.952	0.943	0.943
Altria	0.213	0.212	0.212	0.214	0.212	0.212	0.277	0.277	0.314
American Express	0.722	0.721	0.730	0.723	0.807	1.029	0.375	0.375	0.539
American Intl Group	4.359	4.298	4.475	4.892	5.364	5.723	1.733	1.733	2.186
AT&T	0.700	0.703	0.703	0.702	0.703	0.702	0.326	0.326	0.497
Boeing	0.118	0.117	0.112	0.117	0.118	0.117	0.324	0.274	0.274
Caterpillar	0.782	0.809	0.791	0.791	0.745	0.754	0.385	0.386	0.518
Citigroup	0.634	0.635	0.632	0.634	0.635	0.635	0.350	0.364	0.488
du Pont	2.244	2.266	2.174	2.181	2.119	2.081	0.855	0.784	0.784
Exxon Mobile	0.216	0.218	0.216	0.218	0.460	0.217	0.317	0.317	0.317
GE	0.846	0.863	0.844	0.849	0.854	0.844	0.331	0.331	0.340
GM	6.414	6.205	6.663	5.911	6.795	6.548	2.045	1.972	3.205
Hewlett-Packard	2.732	2.721	2.639	2.704	2.684	2.663	1.163	1.120	1.545
Honeywell	0.007	0.007	0.007	0.007	0.007	0.007	0.145	0.145	0.173
IBM	0.362	0.369	0.361	0.362	0.361	0.361	0.309	0.309	0.309
Intel	2.572	2.468	2.567	2.515	2.322	2.559	0.920	0.961	0.961
Johnson & Johnson	7.482	7.256	7.513	7.516	6.967	7.257	2.540	2.540	3.480
J. P. Morgan	0.818	0.820	0.818	0.817	0.818	0.816	0.366	0.366	0.366
McDonalds	1.862	1.846	1.893	1.886	1.850	1.873	0.574	0.574	0.868
Merck	0.519	0.518	0.516	0.525	0.517	0.519	0.321	0.321	0.389
Microsoft	0.312	0.308	0.309	0.326	0.309	0.309	0.230	0.230	0.326
Pfizer	1.989	1.895	1.815	1.859	3.029	1.830	1.014	1.014	1.343
Coca Cola	1.471	1.524	1.730	1.995	1.572	1.771	0.614	0.614	0.839
Home Depot	1.853	1.923	1.951	2.013	2.117	2.821	0.786	0.786	0.862
Procter & Gamble	0.372	0.377	0.825	0.434	0.367	0.389	0.280	0.280	0.280
United Technologies	2.685	2.686	2.685	2.693	2.685	2.685	0.867	0.862	1.415
Verizon	0.668	0.668	0.669	0.669	0.666	0.667	0.268	0.280	0.389
WalMart	2.611	2.612	2.597	2.691	2.597	2.650	1.030	1.030	1.314
Walt Disney	0.030	0.030	0.030	0.030	0.030	0.030	0.160	0.160	0.160
average	1.681	1.663	1.739	1.718	1.749	1.797	0.693	0.687	0.887

Table 1. Payoffs of LSPI_mle, LSPI_garch, LSPI_data, TVR_mle, TVR_garch, TVR_data, LSM_mle, LSM_garch, and LSM_data, for American put stock options of Dow Jones 30 companies, with semi-annual maturity. Interest rate $r = 0.03$. 500 sample paths are composed for the discovery of exercise policies. The results are averaged over 450 testing paths.

	LSPI			TVR			LSM		
maturity	mle	garch	data	mle	garch	data	mle	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 2. Average payoffs of LSPI, TVR and LSM on real data for Dow Jones 30 companies, with quarterly, semi-annual (repeated from Table 1) and annual maturities.

Now we proceed with the case in which model parameters are estimated from real data. For each company, after estimating parameters for either the GBM model or the SV model from real data, we generate 50,000 sample paths with these parameters. LSPI, TVR and LSM discover the exercise policies with these sample paths. For each company, we evaluate the performance of the discovered policies on 10,000 testing paths, generated with the estimated parameters. The initial stock price in each of the sample path and each of the testing path is set as the first price in the time series of the company.

For each of the Dow Jones 30 companies, we average payoffs over 10,000 testing paths. Then we average the average payoffs over the 30 companies. The results in Table 3 show that LSPI and TVR gain larger payoffs than LSM, both in the GBM model and in the SV model, with interest rate $r = 0.03$.

maturity term	GBM model			SV model		
	LSPI	TVR	LSM	LSPI	TVR	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 3. Average payoffs on synthetic data with parameters estimated from real data.

Again, an explanation for that LSPI and TVR gain larger payoffs is that LSPI and TVR optimize weights across all time steps, whereas LSM makes a single backward pass through time. LSPI and TVR follow the policy iteration approach, so that the policies they discover improve iteratively. LSM learns the policy only once in the backward-recursive approach with least squares regression.

We also vary various parameters for either the GBM or the SV model to generate synthetic sample paths. We vary the interest rate r from 0.01, 0.03 to 0.05, and set the strike price K (initial stock price) to 50. With GBM, we vary the constant volatility σ from 0.1, 0.3 to 0.5. With the SV model, we vary β from 0.2, 0.5 to 0.8, and set $\alpha = 0.96 - \beta$. We test the learned policies on testing paths generated with the same model and the same parameters. Results in Table 4 and Table 5 show that LSPI and TVR outperform or have similar performance as LSM in our studied experiments.

In Figure 1, we present the exercise boundaries discovered by LSPI, TVR and LSM. The optimal exercise boundary for an American put option is a monotonic increasing function, as shown in [6]. Figure 1 (a) for real data from Intel shows that the exercise boundaries discovered by LSPI and TVR are smooth and respect the monotonicity, but not the boundary discovered by LSM. The scarcity of sample paths may explain this non-monotonicity. The boundary of TVR is lower than that of LSPI, which explains that TVR gains larger payoffs than LSPI. Figure 1 (b) shows that the exercise boundary discovered by LSPI is smoother and lower than that discovered by LSM. The exercise boundary discovered by TVR is also smooth. It crosses those of LSPI and LSM.

σ	$r = 0.01$			$r = 0.03$			$r = 0.05$		
	LSPI	TVR	LSM	LSPI	TVR	LSM	LSPI	TVR	LSM
0.1	1.294	1.300	1.286	1.095	1.117	1.061	0.925	0.902	0.896
0.3	4.086	4.062	4.095	3.684	3.666	3.679	3.533	3.604	3.504
0.5	6.965	6.798	6.051	6.514	6.521	6.476	6.315	6.274	6.365

Table 4. Average Payoffs of LSPI, TVR and LSM. $K = 50$. Semi-annual maturity. 50,000 training paths and 10,000 testing paths are generated with the GBM model.

β	$r = 0.01$			$r = 0.03$			$r = 0.05$		
	LSPI	TVR	LSM	LSPI	TVR	LSM	LSPI	TVR	LSM
0.2	0.925	0.931	0.350	0.720	0.721	0.299	0.567	0.555	0.257
0.5	1.167	1.172	0.441	0.960	0.959	0.385	0.792	0.798	0.336
0.8	1.449	1.450	0.548	1.236	1.220	0.485	1.078	1.053	0.430

Table 5. Average Payoffs of LSPI, TVR and LSM. $K = 50$. Semi-annual maturity. 50,000 training paths and 10,000 testing paths are generated with the SV model.

maturity	LSPI			TVR			LSM		
	mle	garch	data	mle	garch	data	mle	garch	data
quarterly	1.862	1.905	1.938	1.869	1.871	1.872	0.613	0.613	0.613
semi-annual	2.733	2.785	2.827	2.649	2.626	2.687	0.578	0.578	0.578
annual	4.171	4.188	4.376	4.149	4.275	4.187	0.680	0.661	0.681

Table 6. Average payoffs of LSPI, TVR and LSM on real data. Asian options.

4.5 Results for American Asian call options

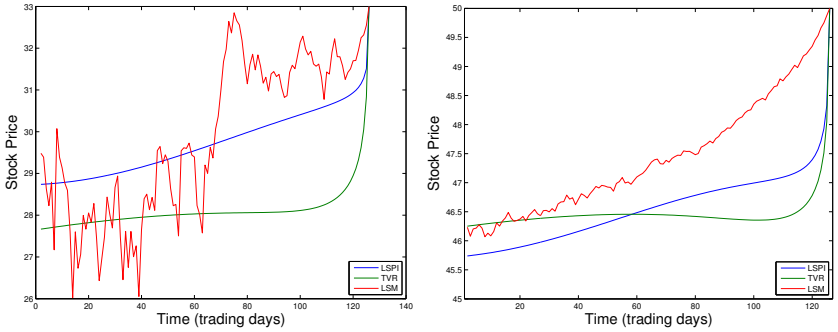
The experimental settings are similar as those for American put options in Sections 4.3 and 4.4. In our experiments, there are 21 lockout days, and the average is taken over the stock prices over the last 21 days.

The experimental results in Table 6 to Table 9 show that LSPI gains larger or similar payoffs than TVR, and both LSPI and TVR gains larger payoffs than LSM. Table 6 shows that for LSPI, policies learned from real data gain larger payoffs than policies learned from simulated samples.

5 Conclusions

Options are important financial instruments, whose prices are usually determined by computational methods. Computational finance is a compelling application area for reinforcement learning research, where hard sequential decision making problems abound and have great practical significance. Our work shows that solution methods developed in reinforcement learning can advance the state of the art in an important and challenging application area, and demonstrates furthermore that computational finance remains an under-explored area for deployment of reinforcement learning methods.

We investigate LSPI for the problem of learning an exercise policy for American options, and compare it with TVR, another policy iteration method, and



(a) Real data for Intel, $r = 0.03$

(b) GBM synthetic data, $r = 0.03$, 50,000 sample paths, $K = S_0 = 50$.

Fig. 1. Exercise boundaries discovered by LSPI, TVR and LSM. Semi-annual maturity.

LSM, the standard least squares Monte Carlo method, on both real and synthetic data. The results show that the exercise policies discovered by LSPI and TVR gain larger payoffs than those discovered by LSM, on both real and synthetic data. Furthermore, for LSPI, TVR and LSM, policies learned from real data generally gain larger payoffs than policies learned from simulated samples. The empirical study shows that LSPI, a solution technique from the reinforcement learning literature, as well as TVR, is superior to LSM, a standard technique from the finance literature, for pricing American options, a classical sequential decision making problem in finance.

It is desirable to investigate alternative reinforcement learning methods, such as the TD method and policy gradient. It is also desirable to investigate more complex models, such as stochastic interest rate models and jump-diffusion models for asset prices and volatility.

References

[1] A. Antos, C. Szepesvari, and R. 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.

[2] D. P. Bertsekas. *Dynamic programming and optimal control*. Athena Scientific, Massachusetts, USA, 1995.

[3] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Massachusetts, USA, 1996.

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

[5] M. Broadie and J. B. Detemple. Option pricing: valuation models and applications. *Management Science*, 50(9):1145–1177, September 2004.

[6] D. Duffie. *Dynamic asset pricing theory*. Princeton University Press, 2001.

[7] P. Glasserman. *Monte Carlo Methods in Financial Engineering*. Springer-Verlag, New York, 2004.

maturity term	GBM model			SV model		
	LSPI	TVR	LSM	LSPI	TVR	LSM
quarterly	2.448	2.323	0.817	2.143	2.044	0.735
semi-annual	3.951	3.897	0.822	3.573	3.377	0.738
annual	6.197	6.030	0.955	5.359	5.128	0.858

Table 7. Average payoffs on simulation data with parameters estimated from real data for Dow Jones 30 companies. Asian options.

σ	$r = 0.01$			$r = 0.03$			$r = 0.05$		
	LSPI	TVR	LSM	LSPI	TVR	LSM	LSPI	TVR	LSM
0.1	1.681	1.427	0.340	1.857	1.630	0.359	2.110	1.630	0.383
0.3	4.694	4.613	1.007	5.064	4.913	1.027	5.340	4.913	1.058
0.5	7.980	7.796	1.702	6.721	7.886	1.722	8.400	7.886	1.700

Table 8. Average Payoffs of LSPI, TVR and LSM. $K = 50$. Semi-annual maturity. 50,000 training paths and 10,000 testing paths, GBM model. Asian options.

β	$r = 0.01$			$r = 0.03$			$r = 0.05$		
	LSPI	TVR	LSM	LSPI	TVR	LSM	LSPI	TVR	LSM
0.2	1.068	1.018	0.214	1.347	1.269	0.235	1.614	1.540	0.257
0.5	1.342	1.234	0.278	1.607	1.489	0.299	1.881	1.759	0.321
0.8	1.697	1.516	0.361	1.915	1.755	0.381	2.172	2.009	0.402

Table 9. Average Payoffs of LSPI, TVR and LSM. $K = 50$. Semi-annual maturity. 50,000 training paths and 10,000 testing paths, SV model. Asian options.

[8] J. C. Hull. *Options, Futures and Other Derivatives (6th edition).* Prentice Hall, 2006.

[9] M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107 – 1149, December 2003.

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

[11] J. Moody and M. Saffell. Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks*, 12(4):875–889, July 2001.

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

[13] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[14] J. N. Tsitsiklis and B. 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.