Extended Abstract: Using Reinforcement Learning in Applied Real Options Modelling

Yuri Lawryshyn*

Keywords: Real Options; Reinforcement Learning; American Options; Exercise Boundary Fitting; Project Valuation.

1 Introduction

Real option analysis (ROA) is recognized as a superior method to quantify the value of real-world investment opportunities where managerial flexibility can influence their worth, as compared to standard discounted cash-flow methods typically used in industry. The ability for managers to react to uncertainties at a future time adds value to projects, and since this value is not captured by standard DCF methods, erroneous decision making may result (Trigeorgis (1996)). A comprehensive ROA of an oil, gas or mineral mining project can improve the allocation of capital and managerial decision making and the methodology is currently used, to some degree, in the commodity extraction sectors. However, realistic models that try to account for a number of risk factors can be mathematically complex, and in situations where many future outcomes are possible, many layers of analysis may be required. Typically, managers are usually unable to understand the models and dismiss results that seem unintuitive to them.

An excellent empirical review of ex-post investment decisions made in copper mining showed that fewer than half of investment timing decisions were made at the right time and 36 of the 51 projects analyzed should have chosen an extraction capacity of 40% larger or smaller (Auger and Guzman (2010)). The authors were unaware of any mining firm basing all or part of their decision making on the systematic use of ROA and emphasize that the "failure to use ROA to assess investments runs against a basic assumption of neoclassical theory: under uncertainty, firms ought to maximize their expected profits". They make the case that irrational decision making exists within the industry due to a lack of real option tools available for better analysis. A number of surveys across industries have found that the use of ROA is in the range of 10-15% of companies, and the main reason for lack of adoption is model complexity (Hartmann and Hassan (2006), Block (2007), Truong, Partington, and Peat (2008), Bennouna, Meredith, and Marchant (2010), Dimitrakopoulos and Abdel Sabour (2007)).

Previously, we introduced a methodology based on exercise boundary fitting (EBF) in an effort to develop a practical Monte Carlo simulation-based real options approach (Bashiri, Davison, and Lawryshyn (2018)). We showed that our methodology converges in the case of simple Bermudan and American put options. More recently, we expanded on the model to solve a staged manufacturing problem (Fleten, Kozlova, and Lawryshyn (2021)). As we presented, utilizing boundary fitting

*Centre for Management of Technology and Entrepreneurship, Faculty of Engineering, University of Toronto, e-mail: yuri.lawryshyn@utoronto.ca

allowed us to solve a computationally difficult problem. In another study we explored the use of the EBF methodology for a number of cases, one being a build and abandon mining example (Davison and Lawryshyn (2021)). We showed that while the methodology provided good convergence on option value, under certain scenarios, where the optimal exercise boundaries occurred in regions where there were few Monte Carlo paths, the optimization algorithm struggled to converge. In Davison and Lawryshyn (2022) we explored convergence issues associated with the methodology and for the build and abandon mining example, we showed that by utilizing heuristic non-convex optimization, namely genetic algorithm (GA), we were able to circumvent the convergence issues, achieving satisfactory results.

Our theoretical and numerical presentation of the EBF method shows how the complexity can be overcome through the use of Monte Carlo simulation and feel that the EBF methodology is very tractable in an industry setting for it is simple enough for managers to understand, yet can account for important real world factors that make the real options model suitable for valuation. However, we recognize that in an effort to account for real world complexities, multiple stochastic factors will need to be modelled. In such cases, the exercise boundaries will be multi-dimensional hyper surfaces. Modelling such surfaces will have its own challenges and will further tax the optimization required with the EBF method. A promising solution to the problem may be the use of reinforcement learning (RL). The purpose of this paper is to explore the opportunity to use RL as a substitute to the EBF method. We note that this study is a work in progress.

The rest of this paper is organized as follows. In the following section we provide a brief review of the literature. We first consider real options in the mining context, as we see this specific application an excellent test case for developing practical multi-factor real real option valuation methods, and then we present a brief review of the application of RL in option valuation. In Section 3 we present our methodology, briefly summarizing our work related to the EBF methodology and then presenting our RL framework. We present our preliminary results in Section 4, and discussion and conclusions in Section **??**.

2 Relevant Literature

Mining Context

The academic literature is very rich in the field of mining valuation. Mining projects are laced with uncertainty and many discounted cash-flow (DCF) methods have been proposed in the literature to try to account for the uncertainty (Bastante, Taboada, Alejano, and Alonso (2008), Dimitrakopoulos (2011), Everett (2013), Ugwuegbu (2013)). Several guidelines/codes have been developed to standardize mining valuation (CIMVAL (2003), VALMIN (2015)). The main mining valuation approaches are income (i.e. cash-flows), market or cost based and the focus of this paper is on income-based real option valuation, which resemble American (or Bermudan) type financial options. Earlier real option works focused on modelling price uncertainty only (Brennan and Schwartz (1985), Dixit and Pindyck (1994), Schwartz (1997)), however the complexity in mining is significant and there are numerous risk factors. Simpler models based on lattice and finite difference methods (FDM) are difficult to implement in a multi-factor setting (Longstaff and Schwartz (2001)) and, also, it is extremely difficult to account for time dependent costs with multiple decision making points (Dimitrakopoulos and Abdel Sabour (2007)). Nevertheless, the simpler models continue to merit attention (Haque, Topal, and Lilford (2014), Haque, Topal, and Lilford (2016)). Dimitrakopoulos and

Abdel Sabour (2007) utilize a multi-factor least squares Monte Carlo (LSMC) approach to account for price, foreign exchange and ore body uncertainty under multiple pre-defined operating scenarios (states). However, the model only allows for operation and irreversible abandonment — aspects such as optimal build time, expansion and mothballing are not considered. Similarly, Mogi and Chen (2007) use ROA and the method developed by Barraquand and Martineau (2007) to account for multiple stochastic factors in a four-stage gas field project. Abdel Saboura and Poulin (2010) develop a multi-factor LSMC model for a single mine expansion.

A review of 92 academic works found that most real options research is focused on dealing with very specific situations where usually no more than two real options are considered (Savolainen (2016)). While the LSMC allows for a more realistic analysis, methods presented to date are applicable only for the case where changes from one state to another does not change the fundamental stochastic factors with time. For example, modular expansion would be difficult to implement in such a model if the cost to expand was a function of time and impacts extracted ore quality due to the changing rate of extraction – these issues were considered in Davison, Lawryshyn, and Zhang (2015) and Kobari, Jaimungal, and Lawryshyn (2014). Also, modelling of multiple layers is still complex and will not lead to a methodology that managers can readily utilize.

Option Valuation Using Reinforcement Learning

Several recent papers apply RL to price/hedge financial options. Buehler, Teichmann, and Wood (2019) used RL and deep neural networks (NN) to approximate an optimal hedging strategy of a portfolio of derivatives considering market frictions. Their model outperformed simple delta hedging on a call option of the S&P500 index. Cannelli, Nuti, Sala, and Szehr (2022) formulated the optimal hedging problem as a risk-averse contextual k-armed bandit problem and showed that their model outperforms deep Q-networks (DQN) in terms of sample efficiency and hedging error when compared to delta hedging on simulated data. Cao, Chen, Hull, and Poulos (2021) used Q-learning and deep deterministic policy gradient (DDPG) to hedge a short position in a call option with transaction costs and showed reduced hedging costs compared to delta hedging on simulated data. Recent other studies applied several different RL models on simulated data and also showed superior performance to delta hedging (Kolm and Ritter (2019), Du, Jin, Kolm, Ritter, Wang, and Zhang (2020)). The application of RL to price options is arguably in its infancy but shows significant promise and may prove to be an excellent solution methodology for complex RO with multiple factors.

3 Methodology

EBF Methodology

In Bashiri, Davison, and Lawryshyn (2018) we presented details of the theory and methodology of applying EBF for the following cases: 1) a Bermudan put option, 2) a Bermudan put option with a variable strike, 3) an American put option and 4) a build / abandon real option. In this section we present the general simulation framework and refer the reader to our previous papers for more details.

The EBF framework assumes that the (real) option valuation is based on a single or multiple stochastic processes, say $\vec{X}_t$, which are simulated using Monte Carlo simulation. Depending on the valuation model, these processes may be risk-neutral or actual and are general in that standard

and non-standard processes can be used. We let $\vec{f}_B(\vec{x}, t; \vec{\theta})$ be a general function that represents N_B exercise boundaries parametrised by $\vec{\theta}$, where $\vec{x}$ represents possible realizations of the process $\vec{X}_t$. We note that $\vec{f}_B(\vec{x}, t; \vec{\theta})$ can be a single point, multiple points, a curve or multiple curves of fixed dimensional surfaces. Based on the path dependent journey of $\vec{X}_t$ we define appropriate first passage of time for the i-th path crossing the j-th boundary as

$$\tau_{B_j}^{(i)} \equiv \min\{t > 0, \vec{X}_t^{(i)} \geq \pm f_{B_j}(\vec{x}, t; \vec{\theta}) \mid \lambda^{(i)}(\vec{X}_t^{(i)})\}, \tag{1}$$

where we use $\pm$ to signify that the process could be hitting the exercise boundary from below or above, depending on the problem at hand, and state $\lambda^{(i)}(\vec{X}_t^{(i)}) \equiv \lambda_t^{(i)}$ with $j \in \{1, 2, ..., N_B\}$, where there could be N_S possible states, also dependent on the problem at hand.

For each simulated path $\vec{X}_t^{(i)}$, we define a cash-flow or payoff at time t as $CF^{(i)}(\vec{X}_t^{(i)}, \lambda_t^{(i)}) \equiv CF_t^{(i)}$ and thus the value generated by the i-th path can be determined by

$$V_0^{(i)}(\vec{X}_t^{(i)}, \lambda_t^{(i)}) \equiv V_0^{(i)} = \sum_{j=0}^{N_t} CF_{t_j}^{(i)} e^{(-rt_j)}, \tag{2}$$

where N_t is the number of time steps in the simulation such that $t \in \{t_0, t_1, ..., t_{N_t}\}$. We emphasize that $V_0^{(i)}$ is a function of $\lambda_t^{(i)}$ and is therefore a function of the exercise boundary parameters, $\vec{\theta}$. The overall option value becomes

$$V_0 = \frac{1}{N} \sum_{i=1}^{N} V_0^{(i)}(\vec{\theta}), \tag{3}$$

where N is the number of paths used in the simulation for $\vec{X}_t$. Our task reduces to maximizing $V_0^{(i)}$ by finding optimal exercise boundary parameters,

$$\vec{\theta}^* = \arg\max_{\vec{\theta}} \frac{1}{N} \sum_{i=1}^{N} V_0^{(i)}(\vec{\theta}) \tag{4}$$

and thus, the option value becomes

$$V_0^* = \frac{1}{N} \sum_{i=1}^{N} V_0^{(i)}(\vec{\theta}^*). \tag{5}$$

Thus, by parametrizing the exercise boundary our solution methodology becomes an optimization of equation (4).

RL Framework

RL is a sub-field of machine learning (ML) where the model automatically learns optimal decisions (actions) over time, typically in a stochastically changing environment. A comprehensive treatment of the topic is provided in Sutton and Barto (2018). As mentioned, this paper is a work in progress and our RL solution methodology will be based on Q-learning, specifically, the model-free framework based on DQN. Since our actions will be discrete for the problems considered here, the DQN framework is appropriate. We explore the application of DQN to value the following scenarios: 1) a Bermudan put option, 2) an American put option, and 3) the build / abandon real option. We will compare the RL results to pseudo-analytical and EBF results. To date, we have not yet been able

to solve the build / abandon problem in a RL framework but hope to have successful results soon. We plan to present some of the challenges faced by the RL framework for this case.

The RL solution framework is formulated within the Markov decision process (MDP) model (again, for details see Sutton and Barto (2018)). The MDP consists of an agent that interacts with its environment and is represented by states, actions and rewards. In the RL framework, the agent acts to optimize expected reward based on the current state of the environment. The agent develops a policy by learning to maximize the total expected reward, given current state and action pairs (Q-learning) by playing multiple games. In the context of real options, each game is represented by a Monte Carlo path of the underlying stochastic factor, or, in the case of a multi-factor setting, the (correlated) paths of the underlying stochastic factors. In the model-free setting, the model and optimal actions are learned through exploration of the environment. In the Deep Q-Network (DQN) framework, a (deep) neural network model is trained to provide actions to optimize the total expected reward for a given state. More sophisticated Q-learning models will be explored in future work.

Bermudan and American Put Options

For the Bermudan and American put options, we consider a GBM stock price process, X_t, as

$$dX_t = rX_t dt + \sigma X_t d\widehat{W}_t, \tag{6}$$

where r is the risk-free rate, σ is the volatility and $\widehat{W}_t$ is a Wiener process in the risk-neutral measure. We assume the payoff of the option to be $\max(K - X_t, 0)$ and can be exercised at times $t_j \in \{t_1, t_2, ..., t_{N_s}, T\}$, $j \in \{1, 2, ..., N_s\}$ where N_s represents the number of steps for which the agent must decide on an action[1]. For the Bermudan option, $N_s = 1$ and we set $t_1 \equiv \tau$. In the RL model, for each episode or game, we generate a two- or mulit-step Monte Carlo simulation. Specifically, for the i-th episode, we have

$$X_{t_j}^{(i)} = X_{t_{j-1}} e^{(r-\sigma^2/2)(t_j - t_{j-1}) + \sigma\sqrt{t_j - t_{j-1}} Z_j^{(i)}}, \tag{7}$$

where $X_{t_0} \equiv X_0$ is the stock price at $t = 0$, $Z_j^{(i)}$ is a standard normal random variable with $i \in \{1, 2, ..., N_e\}$, and N_e is the number of episodes.

The RL reward is modelled as the discounted payoff such that for the j-th step, the reward for the i-th episode is

$$R^{(i)} = \begin{cases} 0, & \text{if action} = 0, \text{ (do not exercise)}, \\ \max\left(K - X_{t_j}^{(i)}, 0\right) e^{-rt_j} & \text{if action} = 1 \text{ (exercise)}. \end{cases} \tag{8}$$

For the Berumudan option, our state space consists of only one variable, namely $S = \{X_\tau\}$ and our action space consists of two discrete actions, at time τ, namely 0 for not exercising and 1 for exercising the option; i.e. $A = \{0, 1\}$. For the American option, our state consists of two variables at each t_j, namely $S_{t_j} = \{X_{t_j}, T - t_j\}$, where at each t_j the agent has the option to exercise or not, thus $A_{t_j} = \{0, 1\}$.

In the DQN formulation, a deep neural network is utilized to estimate the action value function, $Q(S, A)$, where A is the action space. Specifically, the neural network takes as inputs the states and

[1]Note that the agent is not required to take action at $t = T$ as the "exercise" action is automatically dictated by the value of X_T relative to K

produces as output the estimated Q-values for the different actions taken. Thus, for the Bermudan option our network consists of a single input and two outputs whereas for the American option the network consists of two inputs and two outputs.

Build / Abandon Real Option

In the build / abandon real option, we assume the stochastic process, X_t, represents the price of the underlying mineral / material produced or mined. There are four possible states related to the plant, namely,

- plant is not constructed,
- plant is under construction,
- plant is operating,
- plant has been abandoned.

In the RL framework we add two one-hot encoded variables, ξ_1 and ξ_2, where

$$\xi_1 = \begin{cases} 1, & \text{plant is not constructed} \\ 0, & \text{otherwise,} \end{cases}$$

$$\xi_2 = \begin{cases} 1, & \text{plant is operating} \\ 0, & \text{otherwise,} \end{cases}$$

and note that one and only one of $\xi_{i,t_j} = 1$ at any time t_j. Furthermore, we do not need a state variable for the case where the plant is under construction nor abandoned. During the time the plant is being constructed we assume abandonment is not possible. Thus, as soon as the agent decides to construct, the episode time jumps ahead by the time to construct, τ_c, and enters the operating phase (state). This assumption ensures that the environment obeys the Markov property, thus ensuring the problem is a MDP, which RL solution methodologies are based on. We note that this assumption is not restrictive since the probability of the underlying process X_t reaching an abandon boundary in the time to construct after hitting the construction boundary is very low under normal circumstances. Thus, our state at time t_j consists of four variables, $S_{t_j} = \{X_{t_j}, T - t_j, \xi_{1,t_j}, \xi_{2,t_j}\}$.

Before construction, the agent must decide whether to do nothing, i.e. continue waiting (not exercise, action = 0) or construct (exercise, action =1). As discussed above, as soon as the decision to construct is made, the episode jumps to the operating state, so again, the agent must decide whether to continue operating (not exercise, action = 0) or abandon (exercise, action =1). Thus, similar to the American put option above, at each time, t_j, as long as we are not in the construction phase, $A_{t_j} = \{0, 1\}$.

The episode is terminated as follows:

- if $t_j >= T - \tau_c$ if the plant is not constructed,
- if action = 1 (abandonment) while the plant is operating ($\xi_{1,t_j} = 1$),
- if $t = T$ if the plant is operating.

At each t_j except during construction the reward is calculated as follows,

$$R_{t_j} = \begin{cases} -C_w \Delta t e^{-rt_j}, & \text{plant is not constructed and action} = 0, \\ -K e^{-rt_j}, & \text{plant is not constructed and action} = 1, \\ \gamma \left(S_{t_j}^{(i)} - C_o \right) \Delta t e^{-rt_j}, & \text{plant is operating and action} = 0, \\ -C_a e^{-rt_j}, & \text{plant is operating and action} = 1, \\ -C_a e^{-rT}, & \text{plant is operating and } t_j = T, \end{cases} \tag{9}$$

where C_w is the cost rate of waiting (we use $C_w = 0$), $-K$ is the cost of construction, C_o is the operating cost rate, γ is the rate of production (extraction) of the product, Δt is the time step in the simulation and C_a is the cost of abandonment.

Deep Q-Network

In the DQN methodology, a neural network is used to to estimate the action value function, $Q(S, A)$. We used the keras-rl2 library. Our DQN agent was run with with the following parameters:

- the model paremeter, i.e. the neural network, was setup as follows
  - number of inputs was set to one for the Bermudan put option, two for the American put option and four for the build / abandon real option (see above)
  - the neural network was fully connected with varying number of layers and the ReLU (rectified linear unit) activation function was used between layers, except for the last layer leading to the output nodes where a linear activation function was used
  - the output layer consisted of two outputs (as mentioned above, representing $Q(S, A)$ for the two possible actions, do not exercise or exercise)
- the memory parameter was set using the SequentialMemory option with a limit value of 50,000 and window size of 1
- the policy parameter was set to LinearAnnealedPolicy using the EpsGreedyQPolicy, with epsilon set to a maximum value of 1 and decaying to 0.1 over 5000 steps
- the target_model_update was set to the default value of 0.01, so that the target network was updated at a rate of 1% of the total steps taken during training.

As described above, each training episode is based on a single random path for X_t and the number of such paths was varied. In the next section we present preliminary results of our work, exploring the impact of varying certain hyper-parameters such as the neural network size and structure and number of episodes.

4 Results

In this section we summarize our results from previous EBF work and present our preliminary RL results. As will be further discussed below, RL requires significantly longer run times than the EBF, but seems to be better suited to solve the more complicated problems, like the American put and build / abandon real options.

4.1 Bermudan Put Option

As mentioned, we presented the results of using the EBF method in Bashiri, Davison, and Lawryshyn (2018). For the Bermudan option, we used the following parameters:

- $S_0 = 5$
- $K = 5$
- $\tau = 1$
- $T = 2$
- $r = 3\%$
- $\sigma = 10\%$.

For these parameters the pseudo-analytical optimal values are:

- $V_0 = 0.1688$
- $\theta^* = 4.7571$.

In Davison and Lawryshyn (2022) we presented some detailed convergence results for the EBF method, repeated here in Table 1. We note that even if an accurate optimal exercise value, θ^*, is determined, the option value, V_0, will vary depending on the number of simulation paths used. As can be seen in Table 1, we would expect (hope) that the RL agent would learn the optimal exercise value (approximately 3 digits of accuracy) with 100,000 paths.

Table 1: Bermudan put option convergence for 1000 simulation runs; mean value and (standard deviation).

	100 Paths		1000 Paths		10,000 Paths		100,000 Paths		1,000,000 Paths	
V_0	0.1744	(0.0257)	0.1705	(0.0080)	0.1692	(0.0025)	0.1689	(0.0008)	0.1689	(0.0003)
θ^*	4.7052	(0.2324)	4.7353	(0.0819)	4.7538	(0.0334)	4.7564	(0.0157)	4.7572	(0.0070)

For the Bermudan put option, the neural network used to estimate $Q(S, A)$ in the RL consisted of three fully connected hidden layers with 24 nodes per layer leading to 1298 trainable parameters. A histogram plot for θ^* run with 100,000 episodes 40 times is presented in Figure 1. For this set of runs the mean value for θ^* was 4.776 with a standard deviation of 0.034. Compared to the EBF results when 100,000 sample paths were used, we see that the RL results appear to have a potential bias and a higher standard deviation.

4.2 American Put Option

For the American put option we assumed the same parameters for S_0, K, T, r and σ as in Subsection 4.1. As presented in Bashiri, Davison, and Lawryshyn (2018) using a binomial tree, the value of the American put option was determined to be \$0.1835. When simulating X_t we have the freedom to pick both the number of paths N and the number of simulated time steps for each path, N_{step}. To better gauge appropriate values for N and N_{step}, we utilized the exercise boundary determined from the binomial tree and ran 20 simulations for varying N and N_{step}. The mean and standard deviation of each set of 20 simulations is presented in Table 2. As expected, as N and N_{step} were increased, convergence to the true American put option value was achieved. Normally, one would not have

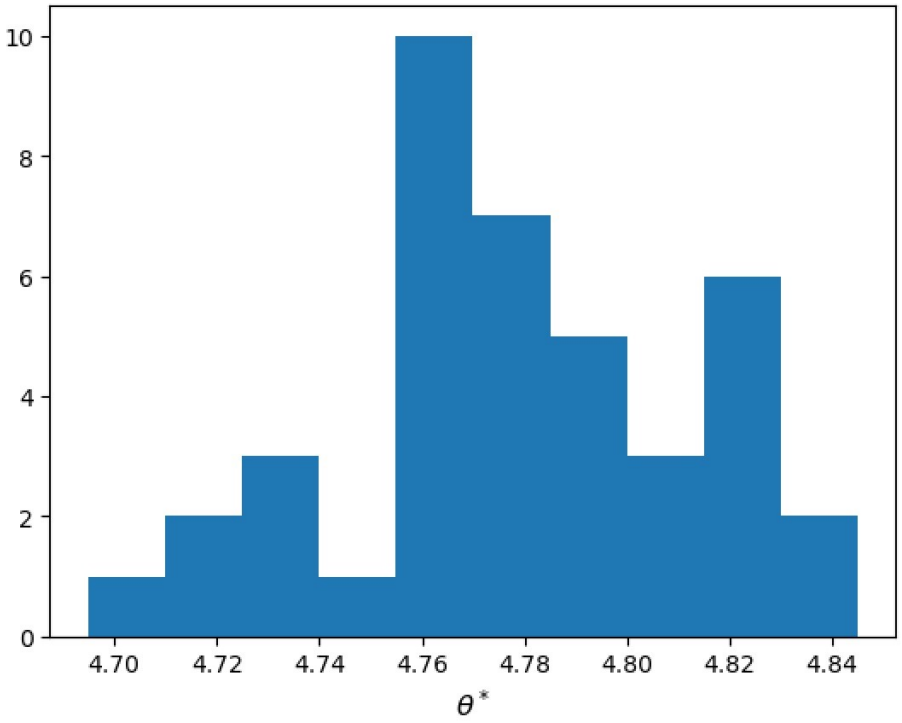

Figure 1: Histograms of θ^* for Bermudan put option using RL with 100,000 episodes run 40 times.

the true exercise boundary a priori and clearly, a number of trial runs would need to be done. For the purpose of presenting the merits of the EBF, we selected $N = 100,000$ and $N_{step} = 500$ for all forthcoming EBF simulations. Using these values for N and N_{step} we creatde a simulation set $\{S_j^{(i)}\}$, where $i \in \{1, 2, 3, ..., N\}$ and $j \in \{1, 2, 3, ..., N_{step}\}$. For this particular simulation set the American put option value was 0.1836 using the true exercise boundary. Naturally, in all forthcoming results we would expect this value to be the upper bound of the American put option values calculated through exercise boundary fitting.

Table 2: Mean and (Standard Deviation) of American Put Option Values with Known Exercise Boundary (20 Simulations)

No. of Steps	No. of Paths (N)							
(N_{step})	1,000		10,000		100,000		500,000	
100	0.1853	(0.00659)	0.1821	(0.00205)	0.1828	(0.00078)	0.1827	(0.00027)
500	0.1836	(0.00654)	0.1831	(0.00260)	0.1833	(0.00066)	0.1834	(0.00026)
1,000	0.1818	(0.00670)	0.1830	(0.00250)	0.1836	(0.00064)	0.1835	(0.00034)

In Davison and Lawryshyn (2019) we explored the use of cubic splines, polynomials and piecewise linear functions to model the exercise boundary. In Figure 2 we plot the results of the optimization using 4 to 8 node cubic splines. Also plotted in the figure are 20 randomly selected paths for X_t. The nodes were spaced evenly over time. As can be seen, while the error in option calculation was less than 5% on average, the exercise boundaries were oscillatory and exhibited significant error.

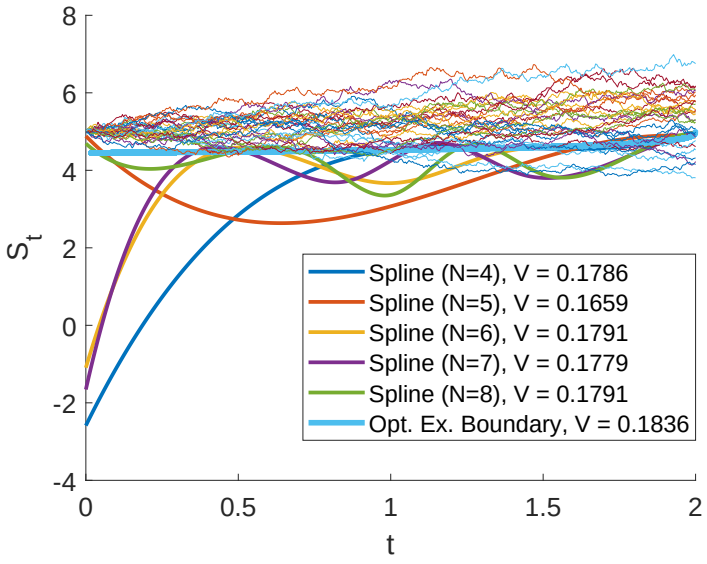

Figure 2: American put option exercise boundary simulation using cubic splines for $h(t;\vec{\eta})$.

Next, we explored the use of second, third and fourth order polynomials for $h(t;\vec{\eta})$. The results are plotted in Figure 3. The average error was approximately 0.5% and the exercise boundaries were close to the optimal boundary. In Figure 4 we plot the results where we assumed a piecewise linear function for $h(t;\vec{\eta})$. As can be seen, as the order increased, the optimal exercise boundary did not necessarily lead to better results.

While not discussed in our previous work, the EBF methodology struggled to converge to an optimal solution when we utilized a Bermudan option methodology with increasing number of exercise options spaced evenly through time. However, the RL American put option methodology is implemented based on this method – namely, by incorporating multiple Bermudan exercise times distributed evenly through time of the life of the option.

For the RL implementation of the American put option, the neural network used to estimate $Q(S, A)$ consisted of three fully connected hidden layers with 48 nodes per layer leading to 4946 trainable parameters. The agent was provided the option to exercise the put option starting at $t = 0.2$ years[2] with 10 evenly spaced times to $T = 2$ years. As a preliminary result, the exercise boundary for one RL solution is plotted in Figure 5. The value of the option was determined to be 0.1776 for this one case.

4.3 Build / Abandon Real Option

Results are being generated

[2]Since there is a very low probability of the X_t process hitting the exercise boundary for low values of t we set the first exercise time to be 10% of T.

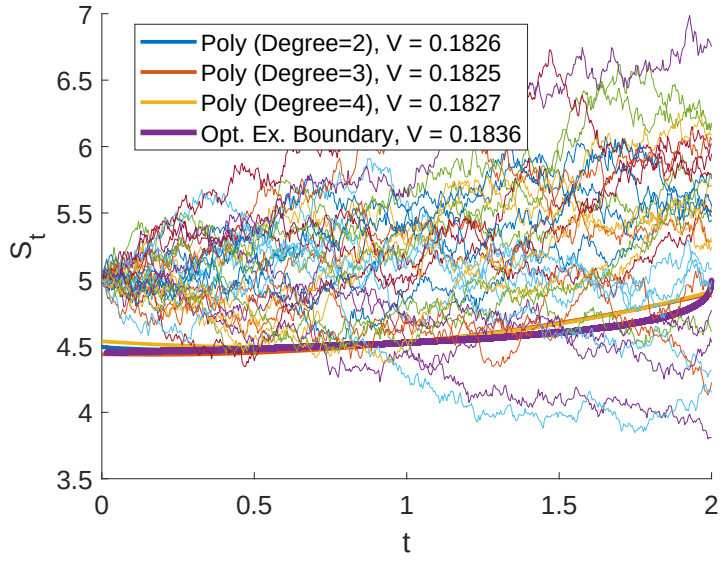

Figure 3: American put option exercise boundary simulation using polynomial functions for $h(t;\vec{\eta})$.

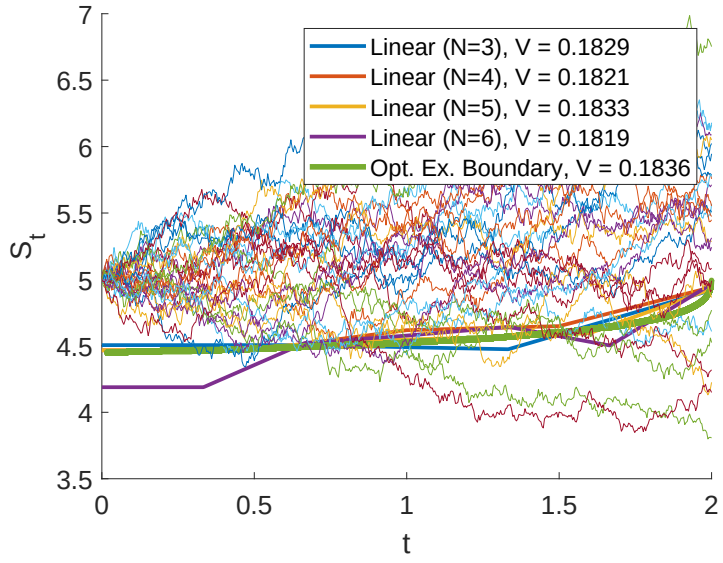

Figure 4: American put option exercise boundary simulation using piecewise linear functions for $h(t;\vec{\eta})$.

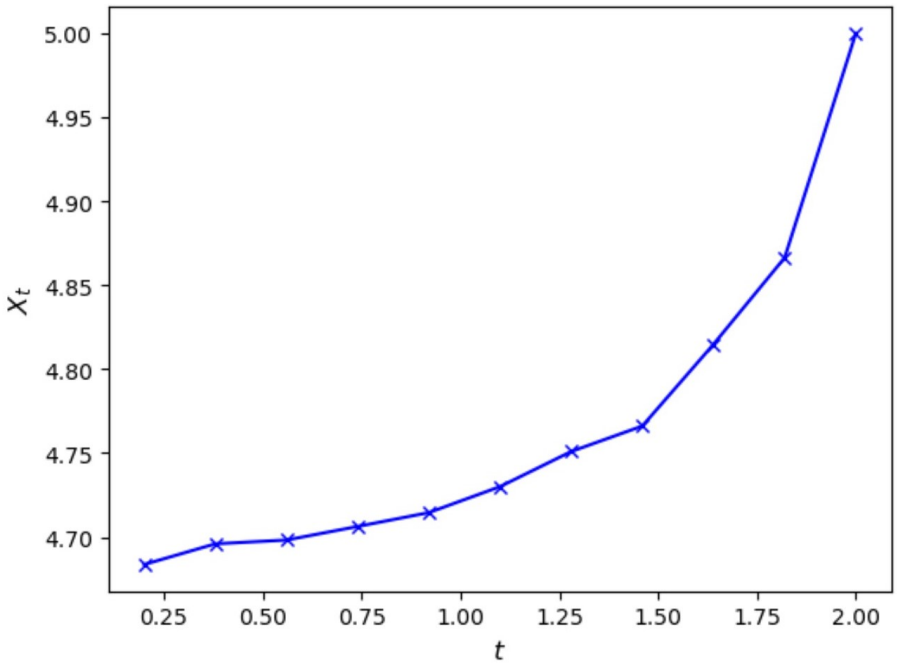

Figure 5: American put option exercise boundary using RL with 100,000 episodes.

5 Conclusions

The focus of our onging research is to develop a real options valuation methodology geared towards practical use. A key innovation of the EBF methodology is the idea of fitting optimal decision making boundaries to optimize the expected value, based on Monte Carlo simulated stochastic processes that represent important uncertain factors. RL provides another framework that shows significant promise. There are a number of advantages that both methods offer. Monte Carlo simulation provides an intuitive platform for managers to understand. Through simulation, complex processes can be utilized and both methods show promise for solving more realistic multi-factor real options problems. When there are fewer optimization variables, the EBF method seems to provide more accurate results than RL. RL shows significant promise in more complicated real options cases where the optimization is dependent on many variables. Drawbacks of RL include significant computational times and a very large set of hyper-parameters that requires tuning.

References

Abdel Saboura, S. and R. Poulin (2010). Mine expansion decisions under uncertainty. *International Journal of Mining, Reclamation and Environment 24*(4), 340–349.

Auger, F. and J. Guzman (2010). How rational are investment decisions in the copper industry? *Resources Policy 35*, 292–300.

Barraquand, J. and D. Martineau (2007). Numerical valuation of high dimensional multivariate american securities. *JOURNAL OF FINANCIAL AND QUANTITATIVE ANALYSIS 30*(3), 383–405.

Bashiri, A., M. Davison, and Y. Lawryshyn (2018). Real option valuation using simulation and exercise boundary fitting - extended abstract. In *Real Options Conference*.

Bastante, F., J. Taboada, L. Alejano, and E. Alonso (2008). Optimization tools and simulation methods for designing and evaluating a mining operation. *Stochastic Environmental Research and Risk Assessment 22*, 727?735.

Bennouna, K., G. Meredith, and T. Marchant (2010). Improved capital budgeting decision making: evidence from canada. *Management Decision 48*(2), 225–247.

Block, S. (2007). Are "real options" actually used in the real world? *Engineering Economist 52*(3), 255–267.

Brennan, M. J. and S. Schwartz (1985). Evaluating natural resource investments. *Journal of Business 58*(2), 135–157.

Buehler, H., G. Teichmann, and B. Wood (2019). Deep hedging. *Quantitative Finance 19*(8), 1271–1291.

Cannelli, L., G. Nuti, M. Sala, and O. Szehr (2022). Hedging using reinforcement learning: Contextual k-armed bandit versus q-learning,. *arXiv*.

Cao, J., J. Chen, J. Hull, and Z. Poulos (2021). Deep hedging of derivatives using reinforcement learning. *The Journal of Financial Data Science 3*, 10–27.

CIMVAL (2003). Standards and guidelines for valuation of mineral properties. Technical report, Canadian Institute of Mining, Metallurgy and Petroleum.

Davison, M. and Y. Lawryshyn (2019). Exercise boundary fitting in real option valuation of complex mining investments. In *Real Options Conference*.

Davison, M. and Y. Lawryshyn (2021). Real option valuation of a mining project using simulation and exercise boundary fitting. In *Canadian Operations Research Conference*.

Davison, M. and Y. Lawryshyn (2022). Convergence of 'exercise boundary fitting? least squares simulation approach. In *Real Options Conference*.

Davison, M., Y. Lawryshyn, and B. Zhang (2015). Optimizing modular expansions in an industrial setting using real options. In *19th Annual International Conference on Real Options*.

Dimitrakopoulos, R. (2011). Stochastic optimization for strategic mine planning: A decade of developments. *Journal of Mining Science 47*(2), 138–150.

Dimitrakopoulos, R. and S. Abdel Sabour (2007). Evaluating mine plans under uncertainty: Can the real options make a difference? *Resources Policy 32*, 116?125.

Dixit, A. and R. Pindyck (1994). *Investment under Uncertainty*. Princeton University Press.

Du, J., M. Jin, P. Kolm, G. Ritter, Y. Wang, and B. Zhang (2020). Deep reinforcement learning for option replication and hedging. *The Journal of Financial Data Science,*, 44?57.

Everett, J. (2013). Planning an iron ore mine: From exploration data to informed mining decisions. *Issues in Informing Science and Information Technology 10*, 145–162.

Fleten, S.-E., M. Kozlova, and Y. Lawryshyn (2021). Real option valuation of staged manufacturing - extended abstract. In *Real Options Conference*.

Haque, M. A., E. Topal, and E. Lilford (2014). A numerical study for a mining project using real options valuation under commodity price uncertainty. *Resources Policy 39*, 115?123.

Haque, M. A., E. Topal, and E. Lilford (2016). Estimation of mining project values through real option valuation using a combination of hedging strategy and a mean reversion commodity price. *Natural Resources Research 25*(4), 459–471.

Hartmann, M. and A. Hassan (2006). Application of real options analysis for pharmaceutical R&D project valuation?empirical results from a survey. *Research Policy 35*, 343–354.

Kobari, L., S. Jaimungal, and Y. A. Lawryshyn (2014). A real options model to evaluate the effect of environmental policies on the oil sands rate of expansion. *Energy Economics 45*, 155–165.

Kolm, P. and G. Ritter (2019). Dynamic replication and hedging: A reinforcement learning approach. *The Journal of Financial Data Science*.

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

Mogi, G. and F. Chen (2007). Valuing a multi-product mining project by compound rainbow option analysis. *International Journal of Mining, Reclamation and Environment 21*(1), 50?64.

Savolainen, J. (2016). Real options in metal mining project valuation: Review of literature. *Resources Policy 50*, 49–65.

Schwartz, E. (1997). The stochastic behaviour of commodity prices: implications for valuation and hedging. *Journal of Finance 52*(3), 923?973.

Sutton, R. and A. Barto (2018). *Reinforcement Learning: An Introduction, Second Edition*. Cambridge, MA: MIT Press.

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

Truong, G., G. Partington, and M. Peat (2008). Cost-of-capital estimation and capital-budgeting practice in australia. *Australian Journal of Management 33*(1), 95–122.

Ugwuegbu, C. (2013). Segilola gold mine valuation using monte carlo simulation approach. *Mineral Economics 26*, 39–46.

VALMIN (2015). The valmin code. Technical report, Australasian Institute of Mining and Metallurgy and the Australian Institute of Geoscientists.