MQLV: Optimal Policy of Money Management in Retail Banking with Q-Learning

Jeremy Charlier[1,2], Gaston Ormazabal[2], Radu State[1], and Jean Hilger[3]

[1] University of Luxembourg, L-1855 Luxembourg, Luxembourg
{name.surname@}@uni.lu
[2] Columbia University, New York NY 10027, USA
{jjc2292,gso7@}@columbia.edu
[3] BCEE, L-1160 Luxembourg, Luxembourg
j.hilger@bcee.lu

Abstract. Reinforcement learning has become one of the best approach to train a computer game emulator capable of human level performance. In a reinforcement learning approach, an optimal value function is learned across a set of actions, or decisions, that leads to a set of states giving different rewards, with the objective to maximize the overall reward. A policy assigns to each state-action pairs an expected return. We call an optimal policy a policy for which the value function is optimal. QLBS, Q-Learner in the Black-Scholes(-Merton) Worlds, applies the reinforcement learning concepts, and noticeably, the popular Q-learning algorithm, to the financial stochastic model of Black, Scholes and Merton. It is, however, specifically optimized for the geometric Brownian motion and the vanilla options. Its range of application is, therefore, limited to vanilla option pricing within the financial markets. We propose MQLV, Modified Q-Learner for the Vasicek model, a new reinforcement learning approach that determines the optimal policy of money management based on the aggregated financial transactions of the clients. It unlocks new frontiers to establish personalized credit card limits or bank loan applications, targeting the retail banking industry. MQLV extends the simulation to mean reverting stochastic diffusion processes and it uses a digital function, a Heaviside step function expressed in its discrete form, to estimate the probability of a future event such as a payment default. In our experiments, we first show the similarities between a set of historical financial transactions and Vasicek generated transactions and, then, we underline the potential of MQLV on generated Monte Carlo simulations. Finally, MQLV is the first Q-learning Vasicek-based methodology addressing transparent decision making processes in retail banking.

Keywords: Q-Learning · Monte Carlo · Payment Transactions.

1 Introduction

A major goal of the reinforcement learning (RL) and Machine Learning (ML) community is to build efficient representations of the current environment to

solve complex tasks. In RL, an agent relies on multiple sensory inputs and past experience to derive a set of plausible actions to solve a new situation [1]. While the initial idea around RL is not new [2–4], significant progress has been achieved recently by combining neural networks and Deep Learning (DL) with RL. The progress of DL [5, 6] has allowed the development of a novel agent combining RL with a class of deep artificial neural networks [1, 7] resulting in Deep Q Network (DQN). The Q refers to the Q-learning algorithm introduced in [8]. It is an incremental method that successively improves its evaluations of the quality of the state-action pairs. The DQN approach achieves human level performance on Atari video games using unprocessed pixels as inputs. In [9], deep RL with double Q-Learning was proposed to challenge the DQN approach while trying to reduce the overestimation of the action values, a well-known drawback of the Q-learning and DQN methodologies. The extension of the DQN approach from discrete to continuous action domain, directly from the raw pixels to inputs, was successfully achieved for various simulated tasks [10].

Nonetheless, most of the proposed models focused on gaming theory and computer game simulation and very few to the financial world. In QLBS [11], a RL approach is applied to the Black, Scholes and Merton financial framework for derivatives [12,13], a cornerstone of the modern quantitative finance. In the BSM model, the dynamic of a stock market is defined as following a Geometric Brownian Motion (GBM) to estimate the price of a vanilla option on a stock [14]. A vanilla option is an option that gives the holder the right to buy or sell the underlying asset, a stock, at maturity for a certain price, the strike price. QLBS is one of the first approach to propose a complete RL framework for finance. As mentioned by the author, a certain number of topics are, however, not covered in the approach. For instance, it is specifically designed for vanilla options and it fails to address any other type of financial applications. Additionally, the initial generated paths rely on the popular GBM but there exist a significant number of other popular stochastic models depending on the market dynamics [15].

In this work, we describe a RL approach tailored for personal recommendation in retail banking regarding money management to be used for loan applications or credit card limits. The method is part of a banking strategy trying to reduce the customer churn in a context of a competitive retail banking market. We rely on the Q-learning algorithm and on a mean reverting diffusion process to address this topic. It leads ultimately to a fitted Q-iteration update and a model-free and off-policy setting. The diffusion process reflects the time series observed in retail banking such as transaction payments or credit card transactions. Such data is, however, strictly confidential and protected by the regulators, and therefore, it cannot be released publicly. Furthermore, we introduce a new terminal digital function, Π, defined as a Heaviside step function in its discrete form for a discrete variable $n \in \mathbb{R}$. The digital function is at the core of our approach for retail banking since it can evaluate the future probability of an event including, for instance, the future default probability of a client based on his spendings.

Our method converges to an optimal policy, and to optimal sets of actions and states, respectively the spendings and the available money. The retail banks can, consequently, determine the optimal policy of money management based on the aggregated financial transactions of the clients. The banks are able to compare the difference between the MQLV's optimal policy and the individual policy of each client. It contributes to an unbiased decision making process while offering transparency to the client. Our main contributions are summarized below:

- A new RL framework called MQLV, Modified Q-Learning for Vasicek, extending the initial QLBS framework [11]. MQLV uses the theoretical foundation of RL learning and Q-Learning to build a financial RL framework based on a mean reverting diffusion process, the Vasicek model [16], to simulate data, in order to reach ultimately a model-free and off-policy RL setting.

- The definition of a digital function to estimate the future probability of an event. The aim is to widen the application perspectives of MQLV by using a characteristic terminal function that is usable for a decision making process in retail banking such as the estimation of the default probability of a client.

- The first application of Q-learning to determine the clients' optimal policy of money management in retail banking. MQLV leverages the clients aggregated financial transactions to define the optimal policy of money management, targeting the risk estimation of bank loan applications or credit cards.

The paper is structured as follows. In section 2, we review QLBS and the Q-Learning formulations derived by Halperin in [11] in the context of the Black, Scholes and Merton model. In section 3, we describe MQLV according to the Q-Learning algorithm that leads to a model-free and off-policy setting. We highlight experimental results in section 4. We discuss related works in section 5 and we conclude in section 6 by addressing promising directions for future work.

2 Background

We define $A_t \in \mathcal{A}$ the action taken at time t for a given state $X_t \in \mathcal{X}$ and the immediate reward by R_{t+1}. The ongoing state is denoted by $X_t \in \mathcal{X}$ and the stochastic diffusion process by $S_t \in \mathcal{S}$ at time t. The discount factor that trades off the importance of immediate and later rewards is expressed by $\gamma \in [0; 1]$.

We recall a policy is a mapping from states to probabilities of selecting each possible action [17]. By following the notations of [11], the policy π such that

$$\pi : \{0, \ldots, T-1\} \times \mathcal{X} \to \mathcal{A} \tag{1}$$

maps at time t the current state $X_t = x_t$ into the action $a_t \in \mathcal{A}$.

$$a_t = \pi(t, x_t) \tag{2}$$

The value of a state x under a policy π, denoted by $v_\pi(x)$ when starting in x and following π thereafter, is called the state-value function for policy π.

$$v_\pi = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | X_t = x \right] \tag{3}$$

The action-value function, $q_\pi(x, a)$ for policy π defines the value of taking action a in state x under a policy π as the expected return starting from x, taking the action a, and thereafter following policy π.

$$q_\pi(x, a) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | X_t = x, A_t = a \right] \tag{4}$$

The optimal policy, π_t^*, is the policy that maximizes the state-value function.

$$\pi_t^*(X_t) = \arg\max_\pi V_t^\pi(X_t) \tag{5}$$

The optimal state-value function, V_t^*, satisfies the Bellman optimality equation such that

$$V_t^*(X_t) = \mathbb{E}_t^{\pi^*} \left[R_t(X_t, u_t = \pi_t^*(X_t), X_{t+1}) + \gamma V_{t+1}^*(X_{t+1}) \right]. \tag{6}$$

The Bellman equation for the action-value function, the Q-function, is defined as

$$Q_t^\pi(x, a) = \mathbb{E}_t \left[R_t(X_t, a_t, X_{t+1}) | X_t = x, a_t = a \right] + \gamma \mathbb{E}_t^\pi \left[V_{t+1}^\pi(X_{t+1}) | X_t = x \right]. \tag{7}$$

The optimal action-value function, Q_t^*, is obtained for the optimal policy with

$$\pi_t^* = \arg\max_\pi Q_t^\pi(x, a). \tag{8}$$

The optimal state-value and action-value functions are connected by the following system of equations.

$$\begin{cases} V_t^* = \max_a Q^*(x, a) \\ Q_t^* = \mathbb{E}_t \left[R_t(X_t, a, X_{t+1}) \right] + \gamma \mathbb{E}_t \left[V_{t+1}^*(X_{t+1} | X_t = x) \right] \end{cases} \tag{9}$$

Therefore, we can obtain the Bellman optimality equation.

$$Q_t^*(x, a) = \mathbb{E}_t \left[R_t(X_t, a_t, X_{t+1}) + \gamma \max_{a_{t+1} \in \mathcal{A}} Q_{t+1}^*(X_{t+1}, a_{t+1}) | X_t = x, a_t = a \right] \tag{10}$$

Using the Robbins-Monro update [18], the update rule for the optimal Q-function with on-line Q-learning on the data point $(X_t^{(n)}, a_t^{(n)}, R_t^{(n)}, X_{t+1}^{(n)})$ is expressed by the following equation with α a constant step-size parameter.

$$\begin{aligned} Q_t^{*,k+1}(X_t, a_t) =& (1-\alpha^k) Q_t^{*,k}(X_t, a_t) + \\ & \alpha^k \left[R_t(X_t, a_t, X_{t+1}) + \gamma \max_{a_{t+1} \in \mathcal{A}} Q_{t+1}^{*,k}(X_{t+1}, a_{t+1}) \right] \end{aligned} \tag{11}$$

3 Algorithm

We describe, in this section, how to derive a general recursive formulation for the optimal action. It is equivalent to an optimal hedge under a financial framework such as, for instance, portfolio or personal finance optimization. We additionally present the formulation of the action-value function, the Q-function. Both the optimal hedge and the Q-function follow the assumption of a continuous space scenario generated by the Vasicek model with Monte Carlo simulation.

By relying on the financial framework established in [11], we consider a mean reverting diffusion process, also known as the Vasicek model [16].

$$dS_t = \kappa(b - S_t)dt + \sigma dB_t \tag{12}$$

The term κ is the speed reversion, b the long term mean level, σ the volatility and B_t the Brownian motion. The solution of the stochastic equation is equal to

$$S_t = S_0 e^{-\kappa t} + b(1 - e^{-\kappa t}) + \sigma e^{-\kappa t} \int_0^t e^{\kappa s} dB_s. \tag{13}$$

Therefore, we define a new time-uniform state variable, i.e. without a drift, as

$$\begin{cases} S_t = X_t + S_0 e^{-\kappa t} + b(1 - e^{-\kappa t}) \\ \text{with } X_t = \sigma e^{-\kappa t} \int_0^t e^{\kappa s} dB_s - [S_0 e^{-\kappa t} + b(1 - e^{-\kappa t})] \end{cases} . \tag{14}$$

Instead of estimating the price of a vanilla option as proposed in [11], we are interested to estimate the future probability of an event using the Q-learning algorithm and a digital function. First, we define the terminal condition reflecting that with the following equation

$$Q_T^*(X_T, a_T = 0) = -\Pi_T - \lambda Var\left[\Pi_T(X_T)\right] \tag{15}$$

where Π_T is the digital function at time $t = T$ defined such that

$$\Pi_T = 1_{S_T \geq K} = \begin{cases} 1 \text{ if } S_T \geq K \\ 0 \text{ otherwise} \end{cases} \tag{16}$$

and the second term, $\lambda Var\left[\Pi_T(X_T)\right]$, is a regularization term with $\lambda \in \mathbb{R}^+ \ll 0$. We use a backward loop to determine the value of Π_t for $t = T - 1, ..., 0$.

$$\Pi_t = \gamma\left(\Pi_{t+1} - a_t \Delta S_t\right) \quad \text{with} \quad \Delta S_t = S_{t+1} - \frac{S_t}{\gamma} = S_{t+1} - e^{r\,\Delta t} S_t \tag{17}$$

Following the definition of the equations (6) and (17), we express the one-step time dependent random reward with respect to the cross-sectional information $\mathcal{F}_t$ as follows.

$$\begin{aligned} R_t(X_t, a_t, X_{t+1}) &= \gamma a_t \Delta S_t(X_t, X_{t+1}) - \lambda Var\left[\Pi_t | \mathcal{F}_t\right] \\ &\text{with } Var\left[\Pi_t | \mathcal{F}_t\right] = \gamma^2 \mathbb{E}_t\left[\hat{\Pi}_{t+1}^2 - 2a_t \Delta\hat{S}_t \hat{\Pi}_{t+1} + a_t^2 \Delta\hat{S}_t^2\right] \end{aligned} \tag{18}$$

The term $\Delta\bar{S}_t$ is defined such that $\Delta\bar{S}_t = \frac{1}{N}\Delta S$, $\Delta\widehat{S} = \Delta S - \Delta\bar{S}_t$ and $\hat{\Pi}_{t+1} = \Pi_{t+1} - \bar{\Pi}_{t+1}$ with $\bar{\Pi}_{t+1} = \frac{1}{N}\Pi_{t+1}$. Because of the regularizer term, the expected reward R_t is quadratic in a_t and has a finite solution. Therefore, we inject the one-step time dependent random reward equation (18) into the Bellman optimality equation (10) to obtain the following Q-learning update, Q^*, and the optimal action, a^*, to be solved within a backward loop $\forall t = T-1, ..., 0$.

$$\begin{aligned} Q_t^*(X_t, a_t) &= \gamma \mathbb{E}_t\left[Q_{t+1}^*(X_{t+1}, a_{t+1}^*) + a_t \Delta S_t\right] - \lambda Var\left[\Pi_t | \mathcal{F}_t\right] \\ a_t^*(X_t) &= \mathbb{E}_t\left[\Delta\hat{S}_t \hat{\Pi}_{t+1} + \frac{1}{2\lambda\gamma}\Delta S_t\right]\left[\mathbb{E}_t\left[\left(\Delta\hat{S}_t\right)^2\right]\right]^{-1} \end{aligned} \tag{19}$$

We refer to [11] for further details about the analytical solution, a^*, of the Q-learning update (19). Our approach uses the N Monte Carlo paths simultaneously to determine the optimal action a^* and the optimal action-value function Q^* to learn the policy π^*. Thus, we do not need an explicit conditioning of X_t at time t. We assume a set of basis function $\{\Phi_n(x)\}$ for which the optimal action $a_t^*(X_t)$ and the optimal action-value function, $Q_t^*(X_t, a_t^*)$, can be expanded.

$$a_t^*(X_t) = \sum_n^M \phi_{nt}\Phi_n(X_t) \quad \text{and} \quad Q_t^*(X_t, a_t^*) = \sum_n^M \omega_{nt}\Phi_n(X_t) \tag{20}$$

The coefficients ϕ and ω are computed recursively backward in time $\forall t = T-1, \ldots, 0$. Subsequently, we define the minimization problem to evaluate ϕ_{nt}.

$$G_t(\phi) = \sum_{k=1}^N \left[-\sum_n^M \phi_{nt}\Phi_n(X_t^k)\Delta S_t^k + \gamma\lambda\left(\Pi_{t+1}^k - \sum_n^M \phi_{nt}\Phi_n(X_t^k)\Delta\widehat{S}_t^k\right)^2\right] \tag{21}$$

The equation (21) leads to the following set of linear equations $\forall n = 1, \ldots, M$.

$$\begin{cases} A_{nm}^{(t)} = \displaystyle\sum_{k=1}^N \Phi_n(X_t^k)\Phi_m(X_t^k)(\Delta\widehat{S}_{t^k})^2 \\ B_n^{(t)} = \displaystyle\sum_{k=1}^N \Phi_n(X_t^k)\left[\widehat{\Pi}_{t+1}^k \Delta\widehat{S}_t^k + \frac{1}{2\gamma\lambda}\Delta S_t^k\right] \end{cases} \quad \text{with } \sum_m^M A_{nm}^{(t)}\phi_{mt} = B_n^{(t)} \tag{22}$$

Therefore, the coefficients of the optimal action $a_t^*(X_t)$ is determined by

$$\phi_t^* = A_t^{-1} B_t. \tag{23}$$

Hereinafter, we use Fitted Q Iteration (FQI) [19, 20] to evaluate the coefficients ω. The optimal action-value function, $Q^*(X_t, a_t)$, is represented in its matrix form according to the basis function expansion of the equation (20).

$$\begin{aligned} Q_t^*(X_t, a_t) = &\left(1, a, \frac{1}{2}a_t^2\right) \begin{pmatrix} W_{11}(t)\ W_{12}(t) \dots W_{1M}(t) \\ W_{21}(t)\ W_{22}(t) \dots W_{2M}(t) \\ W_{31}(t)\ W_{32}(t) \dots W_{3M}(t) \end{pmatrix} \begin{pmatrix} \Phi_1(X_t) \\ \vdots \\ \Phi_M(X_t) \end{pmatrix} \\ =&A_t^T W_t \Phi(X_t) = A_t^T U_W(t, X_t) \end{aligned} \tag{24}$$

Based on the least-square optimization problem, the coefficient W_t are determined using backpropagation $\forall t = T-1, ..., 0$ as follows

$$\begin{aligned} \mathcal{L}_t(W_t) = &\sum_{k=1}^{N} \left(R_t(X_t, a_t, X_{t+1}) + \gamma \max_{a_{t+1} \in \mathcal{A}} Q_{t+1}^*(X_{t+1}, a_{t+1}) - W_t \Psi_t(X_t, a_t) \right)^2 \\ &\text{with } W_t \Psi(X_t, a_t) + \epsilon \underset{\epsilon \to 0}{\longrightarrow} R_t(X_t, a_t, X_{t+1}) + \gamma \max_{a_{t+1} \in \mathcal{A}} Q_{t+1}^*(X_{t+1}, a_{t+1}) \end{aligned} \tag{25}$$

for which we derive the following set of linear equations.

$$\begin{cases} M_n^{(t)} = \displaystyle\sum_{k=1}^{N} \Psi_n(X_t^k, a_t^k) \left[\eta \left(R_t(X_t, a_t, X_{t+1}) + \gamma \max_{a_{t+1} \in \mathcal{A}} Q_{t+1}^*(X_{t+1}, a_{t+1}) \right) \right] \\ \text{with } \eta \sim B(N, p) \end{cases} \tag{26}$$

The term $B(N, p)$ represents the binomial distribution for n samples with probability p. It plays the role of a dropout function when evaluating the matrix M_t to compensate the well-known drawback of the Q-learning algorithm that is the overestimation of the Q-function values. We reach finally the definition of the optimal weights to determine the optimal action a^*.

$$W_t^* = S_t^{-1} M_t \tag{27}$$

The proposed model does not require any assumption on the dynamics of the time series, neither transition probabilities nor policy or reward functions. It is an off-policy model-free approach. The computation of the optimal policy, the optimal action and the optimal Q-function that leads to the future event probabilities is summed up in algorithm 1.

Algorithm 1: Q-learning to evaluate the optimal policy of money management

Data: time series of maturity T, either from generated or true data

Result: optimal Q-function Q^*, optimal action a^*, value of digital function Π

begin

  /*Condition at T*/

  $a_T^*(X_T) = 0$

  $Q_T(X_T, a_T) = -\Pi_T = -1_{S_T \geq K}$ using equation (16)

  $Q_T^*(X_T, a_T^*) = Q_T(X_T, a_T)$

  /*Backward Loop*/

  for $t \leftarrow T-1$ **to** 0 **do**

    /*Evaluate the coefficients ϕ*/

    compute A_t, B_t using equation (22)

    $\phi_t^* \leftarrow A_t^{-1} B_t$

    /*Evaluate the coefficients ω*/

    compute S_t, M_t using equation (26)

    $W_t^* \leftarrow S_t^{-1} M_t$

    $a_t^*(X_t) = \sum_n^M \phi_{nt}^* \Phi_n(X_t)$

    $Q^*(X_t, a_t) = A_t^T W_t^* \Phi_(X_t)$

  /*Compute the digital function value to estimate the event probability at $t = 0$*/

  `print(`$\Pi_0 = mean(Q_0^*)$`)`

return

4 Experiments

We empirically evaluate the performance of MQLV. We initially highlight the similarities between historical payment transactions and Vasicek generated transactions. We then underline the MQLV's capabilities to learn the optimal policy of money management based on the estimation of future event probabilities in comparison to the closed formula of [12,13], hereinafter denoted by BSM's closed formula. We rely on synthetic data sets because of the privacy and the confidentiality issues of the retail banking data sets.

Data Availability and Data Description One of our contributions is to bring a RL framework designed for retail banking. However, none of the data sets can be released publicly because of the highly sensitive information they contain. We therefore show the similarities between a small sample of anonymized transactions and Vasicek generated transactions [16]. We then use the Vasicek mean reverting stochastic diffusion process to generate larger synthetic data sets

similar to the original retail banking data sets. The mean reverting dynamic is particularly interesting since it reflects a wide range of retail banking transactions including the credit card transactions, the savings history or the clients' spendings. Three different data sets were generated to avoid any bias that could have been introduced by using only one data set. We choose to differentiate the number of Monte Carlo paths between the data sets to assess the influence of the sampling size on the results. The first, second and third data sets contain respectively 20,000, 30,000 and 40,000 paths. We release publicly the data sets[4] to ensure the reproducibility of the experiments.

Experimental Setup and Code Availability In our experiments, we generate synthetic data sets using the Vasicek model with a parameter $S_0 = 1.0$ corresponding to the value of the time series at $t = 0$, a maturity of six months $T = 0.5$, a speed reversion $a = 0.01$, a long term mean $b = 1$ and a volatility $\sigma = 0.15$. Because the choice of the parameters of the Vasicek model do not have any influence on the results of the Q-learning approach, the numbers were fixed such that any limitations of the methodology would be quickly observed. The number of time steps is fixed equal to 5. We additionally use different strike values for the experiments explicitly mentioned in the Results and Discussions subsection. The simulations were performed on a computer with 16GB of RAM, Intel i7 CPU and a Tesla K80 GPU accelerator. To ensure the reproducibility of the experiments, the code is available at the following address[4].

Results and Discussions about MQLV As aforementioned, we cannot release publicly an anonymized transactions data set because of privacy, confidentiality and regulatory issues. We consequently highlight the similarities between the dynamic of a small sample of anonymized transactions and Vasicek generated transactions for one client [21] in figure 1. The financial transactions in retail banking are periodic and often fluctuates around a long term mean, reflecting the frequency and the amounts of the spendings habits of the clients. The akin dynamic of the original and the generated transactions is highlighted by the small RMSE of 0.03. We also performed a least square calibration of the Vasicek parameters to assess the model's plausibility. We can observe in table 1 that the Vasicek parameters have the same magnitude and, therefore, it supports the hypothesis that the Vasicek model could be used to generate synthetic transactions.

We present the core of our contribution in the following experiment. We aim at learning the optimal policy of money management. It is particularly interesting for bank loan applications where the differences between a client's spendings policy and the optimal policy can be compared. We show that MQLV is capable of evaluating accurately the probability of a default event using a digital function which highlights the learning of the optimal policy of money management. Effectively, if the MQLV's learned policy is different than the optimal policy, then the probabilities of default events are not accurate. In figure 2, the esti-

[4] The code and the data sets are available at https://github.com/dagrate/MQLV.

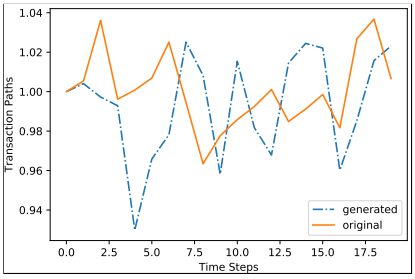

Fig. 1. Samples of original and Vasicek generated transactions for one client. The two samples oscillate around a long term mean of 1 and have a similar pattern, highlighted by the small RMSE of 0.03 in table 1.

Table 1. RMSE error between the samples of original transactions and generated Vasicek transactions of figure 1. We also calibrated the Vasicek parameters according to the original transactions to validate the model's plausibility.

Description	Value
RMSE	0.0335
Vasicek speed reversion a	0.5444
Vasicek long term mean b	0.9001
Vasicek volatility σ	0.2185

mation of future event probabilities for different strike values is represented. We rely on the BSM's closed formula for the vanilla option pricing [12, 13] to approximate the digital function values [15]. We used, therefore, the BSM's values as reference values to cross-validate the MQLV's values. MQLV achieves a close representation of the event probabilities for the different strike values in figure 2. The curves of both the MQLV and the BSM's approaches are similar with a RMSE of 1.5016. This result highlights that the learned Q-learning policy of MQLV is sufficiently close to the optimal policy to compute event probabilities almost identical to the probabilities of the BSM's formula approximation.

We gathered quantitative results in table 2 for a deeper analysis of the MQLV's results. The event probability values are listed for the three data sets. We chose a set of parameters for the Vasicek model such that our configuration is free of any time-dependency. We therefore expect a probability value of 50% at a threshold of 1 because the standard deviation of the generated data sets is only induced by the normal distribution standard deviation, used to simulate the Brownian motion. Surprisingly, the MQLV values at a strike of 1 are closer to 50% than the BSM's values for all the data sets. We can conclude, subsequently, that, for our configuration, MQLV is capable to learn the optimal policy of money management which is reflected by the accurate evaluation of the event probabilities.

We chose to generate three new data sets with new Vasicek parameters a and σ to underline the potential of MQLV and the universality of the results. In table 3, we computed the event probabilities for different strikes for the newly generated data sets. The parameter b remains unchanged since we want to keep a configuration free of any time-dependency. We notice that MQLV is capable to estimate a probability of 50% for a strike of 1 which can only be obtained if MQLV is able to learn the optimal policy. We also observe that the BSM's approximation does lead to a lower accuracy. We showed in this experiment that

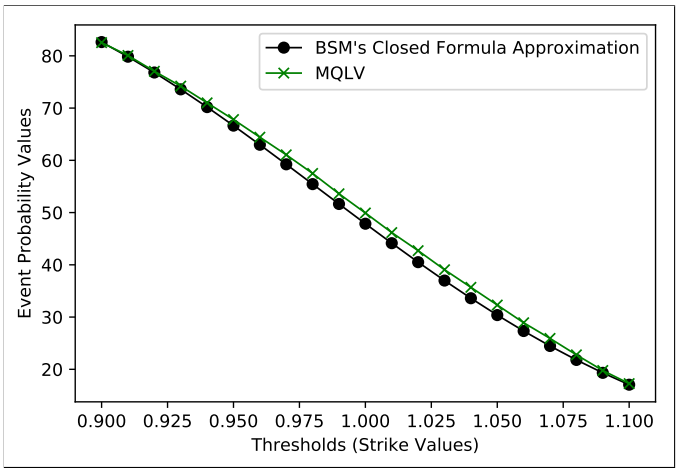

Fig. 2. Event probability values calculated by MQLV and BSM's closed formula approximation for 40,000 Monte Carlo paths with Vasicek parameters $a = 0.01, b = 1$ and $\sigma = 0.15$. The BSM's closed formula approximation values are used as reference values. The event probabilities of MQLV are close to the BSM's values with a total RMSE of 1.502. It illustrates that MQLV is able to learn the optimal policy leading to accurate event probabilities.

Table 2. Valuation differences of the digital values for event probabilities according to different strikes between the BSM's closed formula approximation and MQLV. Given our time-uniform configuration, the event probability values should be close to 50% for a strike value of 1. The MQLV values are close to the theoretical target of 50% at a strike of 1 highlighting the MQLV's capabilities to learn the optimal policy. The BSM's closed formula approximation slightly underestimates the probability values.

Data Set	Number of Paths	Strike Values	BSM's Approx. Values (%)	MQLV Values (%)	Absolute Difference
1	20,000	0.92	76.810	**77.098**	0.288
1	20,000	0.98	55.447	**57.920**	2.473
1	20,000	1.00	47.867	**50.235**	2.368
1	20,000	1.02	40.509	**42.865**	2.356
2	30,000	0.92	76.810	**76.953**	0.143
2	30,000	0.98	55.447	**57.760**	2.313
2	30,000	1.00	47.867	**50.043**	2.176
2	30,000	1.02	40.509	**42.744**	2.235
3	40,000	0.92	76.810	**77.047**	0.237
3	40,000	0.98	55.447	**57.491**	2.044
3	40,000	1.00	47.867	**49.924**	2.057
3	40,000	1.02	40.509	**42.713**	2.204

Table 3. Event probabilities for data sets generated with different Vasicek parameters a and σ. The parameter b remains unchanged to keep a configuration free of any time-dependency to facilitate the results explainability. We can deduce that MQLV is able to learn the optimal policy because the MQLV's probabilities are close to the theoretical target of 50% at a strike of 1. MQLV is also more accurate than BSM's formula.

Parameters $a; b; \sigma$	Number of Paths	Strike Values	BSM's App. Values (%)	MQLV Values (%)	Absolute Difference
0.01; 1; 0.10	50,000	0.98	59.856	**61.223**	1.366
0.01; 1; 0.10	50,000	1.00	48.562	**50.001**	1.439
0.01; 1; 0.10	50,000	1.02	37.596	**39.044**	1.447
0.01; 1; 0.30	50,000	0.98	49.558	**53.647**	4.089
0.01; 1; 0.30	50,000	1.00	45.767	**49.997**	4.230
0.01; 1; 0.30	50,000	1.02	42.088	**46.194**	4.106
0.10; 1; 0.15	50,000	0.98	55.447	**57.540**	2.093
0.10; 1; 0.15	50,000	1.00	47.867	**50.015**	2.148
0.10; 1; 0.15	50,000	1.02	40.509	**42.638**	2.129
0.30; 1; 0.15	50,000	0.98	55.447	**57.586**	2.139
0.30; 1; 0.15	50,000	1.00	47.867	**50.022**	2.155
0.30; 1; 0.15	50,000	1.02	40.509	**42.542**	2.033

our model-free and off-policy RL approach, MQLV, is able to learn the optimal policy reflected by the accurate probability values independently of the data sets considered and of the Vasicek parameters.

Limitations of the BSM's closed formula used for cross validation In our experiments, we observed, surprisingly, that the BSM's closed formula approximation underestimates the event probability values. The volatility is the only parameter playing a significant role in the generation of the time series and, therefore, the event probability should be equal to the mean of the distribution used to generate the random numbers. The Brownian motion is simulated with a standard normal distribution with a 0.5 mean. The BSM's closed formula did not, however, lead to a probability of 0.5 but to slightly smaller values because of the limit of their theoretical framework [12,13]. Hence, we observed that MQLV was more accurate than the BSM's closed formula in our configuration.

5 Related Work

The foundations of modern reinforcement learning described in [2, 4] established the theoretical framework to learn good policies for sequential decision problems by proposing a formulation of cumulative future reward signal. The Q-learning algorithm introduced in [3] is one of the cornerstone of all recent reinforcement learning publications. However, the convergence of the Q-Learning algorithm was solved several years later. It was shown that the Q-Learning algorithm with

non-linear function approximators [22] with off-policy learning [23] could provoke a divergence of the Q-network. Therefore, the reinforcement learning community focused on linear function approximators [22] to ensure convergence.

The emergence of neural networks and deep learning [24] contributed to address the use of reinforcement learning with neural networks. At an early stage, deep auto-encoders were used to extract feature spaces to solve reinforcement learning tasks [25]. Then, thanks to the release of the Atari 2600 emulator [26], a public data set was available answering the needs of the RL community for larger simulation. The Atari emulator allowed a proper performance benchmark of the different reinforcement learning algorithms and offered the possibility to test various architectures. The Atari games were used to introduce the concept of deep reinforcement learning [1, 7]. The authors used a convolutional neural network trained with a variant of Q-learning to successfully learn control policies directly from high dimensional sensory inputs. They reached human-level performance on many of the Atari games. Shortly after, the deep reinforcement learning was challenged by double Q-Learning within a deep reinforcement learning framework [9]. The double Q-Learning algorithm was initially introduced in [19] in a tabular setting. The double deep Q-Learning gave more accurate estimates and lead to much higher scores than the one observed in [1, 7]. Consequently, an ongoing work is to further improve the results of the double deep Q-learning algorithms through different variants. In [27], the authors used a quantile regression to approximate the full quantile function for the state-action return distribution, leading to a large class of risk-sensitive policies. It allowed them to further improve the scores on the Atari 2600 games simulator. Similarly, a new algorithm, called C51, which applies the Bellman's equation to the learning of the approximate value distribution was designed in [28]. They showed state-of-the-art results on the Atari 2600 emulator.

Other publications meanwhile focused on model-free policies and actor-critic framework. Stochastic policies were trained in [29] with a replay buffer to avoid divergence. It was showed in [30] that deterministic policy gradients (DPG) exist, even in a model-free environment. Subsequently, the DPG approach was extended in [31] using a deviator network. Continuous control policies were learned using backpropagation introducing the Stochastic Value Gradient SVG(0) and SVG(1) in [32]. Recently, Deep Deterministic Policy Gradient (DDPG) was presented in [10] to learn competitive policies using an actor-critic model-free algorithm based on the DPG that operates over continuous action spaces.

6 Conclusion

We introduced Modified Q-Learning for Vasicek or MQLV, a new model-free and off-policy reinforcement learning approach capable of evaluating an optimal policy of money management based on the aggregated transactions of the clients.

MQLV is part of a banking strategy that looks to minimize the customer churn by including more transparency and more personalization in the decision process related to bank loan applications or credit card limits. It relies on a digital function to estimate the future probability of an event such as a payment default. We discuss its relation with the Bellman optimality equation and the Q-learning update. We conducted experiments on synthetic data sets because of the privacy and confidentiality issues related to the retail banking data sets. The generated data sets followed a mean reverting stochastic diffusion process, the Vasicek model, simulating retail banking data sets such as transaction payments. Our experiments showed the performance of MQLV with respect to the BSM's closed formula for vanilla options. We also highlighted that MQLV is able to determine an optimal policy, an optimal Q-function, optimal actions and optimal states reflected by accurate probabilities. Surprisingly, we observed that MQLV led to more accurate event probabilities than the popular BSM's formula.
Future work will address the creation of a fully anonymized data set illustrating the retail banking daily transactions with a privacy, confidentiality and regulatory compliance. We will also evaluate the MQLV's performance for data sets that violate the Vasicek assumptions. We, furthermore, observed that the Q-learning update could minor the real probability values for simulation involving a small temporal discretization such as $\Delta t = 200$. Preliminary results showed it is provoked by the basis function approximator error. We will address this point in future research. Finally, we will extend the Q-learning update to other scheme for improved accuracy and incorporate a deep learning framework.

References

1. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602 (2013)
2. Sutton, R.S.: Temporal credit assignment in reinforcement learning (1984)
3. Watkins, C.J.C.H.: Learning from delayed rewards. Ph.D. thesis, King's College, Cambridge (1989)
4. Williams, R.: A class of gradient-estimation algorithms for reinforcement learning in neural networks. In: Proceedings of the International Conference on Neural Networks. pp. II–601 (1987)
5. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
6. Sermanet, P., Kavukcuoglu, K., Chintala, S., LeCun, Y.: Pedestrian detection with unsupervised multi-stage feature learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3626–3633 (2013)
7. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529 (2015)
8. Watkins, C.J., Dayan, P.: Q-learning. Machine learning **8**(3-4), 279–292 (1992)
9. Van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning. In: AAAI. vol. 2, p. 5. Phoenix, AZ (2016)

10. Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971 (2015)
11. Halperin, I.: Qlbs: Q-learner in the black-scholes (-merton) worlds. arXiv preprint arXiv:1712.04609 (2017)
12. Black, F., Scholes, M.: The pricing of options and corporate liabilities. Journal of political economy **81**(3), 637–654 (1973)
13. Merton, R.C.: Theory of rational option pricing. The Bell Journal of economics and management science pp. 141–183 (1973)
14. Wilmott, P.: Paul Wilmott on quantitative finance. John Wiley & Sons (2013)
15. Hull, J.C.: Options futures and other derivatives. Pearson Education India (2003)
16. Vasicek, O.: An equilibrium characterization of the term structure. Journal of financial economics **5**(2), 177–188 (1977)
17. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction. MIT press (2018)
18. Robbins, H., Monro, S.: A stochastic approximation method. In: Herbert Robbins Selected Papers, pp. 102–109. Springer (1985)
19. Hasselt, H.V.: Double q-learning. In: Advances in Neural Information Processing Systems. pp. 2613–2621 (2010)
20. Murphy, S.A.: A generalization error for q-learning. Journal of Machine Learning Research **6**(Jul), 1073–1097 (2005)
21. Santander: Santander product recommendation. https://www.kaggle.com/c/santander-product-recommendation/data (2016)
22. Tsitsiklis, J.N., Van Roy, B.: Analysis of temporal-diffference learning with function approximation. In: Advances in neural information processing systems. pp. 1075–1081 (1997)
23. Baird, L.: Residual algorithms: Reinforcement learning with function approximation. In: Machine Learning Proceedings 1995, pp. 30–37. Elsevier (1995)
24. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: Deep learning, vol. 1. MIT press Cambridge (2016)
25. Lange, S., Riedmiller, M.: Deep auto-encoder neural networks in reinforcement learning. In: The 2010 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2010)
26. Bellemare, M.G., Naddaf, Y., Veness, J., Bowling, M.: The arcade learning environment: An evaluation platform for general agents. Journal of Artificial Intelligence Research **47**, 253–279 (2013)
27. Dabney, W., Ostrovski, G., Silver, D., Munos, R.: Implicit quantile networks for distributional reinforcement learning. arXiv preprint arXiv:1806.06923 (2018)
28. Bellemare, M.G., Dabney, W., Munos, R.: A distributional perspective on reinforcement learning. arXiv preprint arXiv:1707.06887 (2017)
29. WawrzyńSki, P., Tanwani, A.K.: Autonomous reinforcement learning with experience replay. Neural Networks **41**, 156–167 (2013)
30. Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M.: Deterministic policy gradient algorithms. In: ICML (2014)
31. Balduzzi, D., Ghifary, M.: Compatible value gradients for reinforcement learning of continuous deep policies. arXiv preprint arXiv:1509.03005 (2015)
32. Heess, N., Wayne, G., Silver, D., Lillicrap, T., Erez, T., Tassa, Y.: Learning continuous control policies by stochastic value gradients. In: Advances in Neural Information Processing Systems. pp. 2944–2952 (2015)