Reinforcement Learning Approach for Solving Asset Allocation Problem

WU, Yuheng
21107083
yuheng.wu@connect.ust.hk

FAN, Kwan Wai
05037383
kwfanaa@connect.ust.hk

***Abstract*—This report presents our solution to the Asset Allocation problem for MAFS5370 / MSBD6000M Assignment 1. We first introduce the mathematical formulation and derive analytical solutions under specific conditions. We then apply simulation and reinforcement learning approaches, primarily using Q-Learning as our TD method. Our study explores different state and action spaces to evaluate solution reliability. Finally, we compare the results from analytical, Monte-Carlo, and RL-based methods, highlighting their advantages and limitations.**

I. Introduction

This report serves as the final submission for **MAFS5370 / MSBD6000M Assignment 1**. In this report, we first describe the Asset Allocation problem we aim to solve and present its mathematical formulation. We then approach the problem using both simulation and reinforcement learning techniques. Specifically, we utilize Q-Learning as our primary Temporal Difference (TD) method. Starting from the simplest version of the problem, we progressively derive more complex formulations, ultimately presenting our conclusions and comparisons.

The structure of this report is as follows:

- **Chapter 2:** We clearly describe the problem we need to solve in this project.
- **Chapter 3:** We introduce the usage of the project code and provide instructions on reproducing the experiments presented in this report.
- **Chapter 4:** We present the mathematical derivations, analyzing the marginal conditions and constraints of the analytical solution.
- **Chapter 5:** We employ Monte-Carlo simulations to obtain numerical solutions.
- **Chapter 6:** We solve the problem using various reinforcement learning approaches. Different assumptions on the action space and state space are tested to evaluate the reliability of observed results.
- **Chapter 7:** We conclude with a summary of findings and discussions.

II. Problem Description

The problem assume a discrete-time asset allocation problem. There are two kind of asset, one is risky and the other is risk free. The risky asset yields a random return with $Y_t \sim a\delta_p + b\delta_{1-p}, a \geq b$, the risk free asset has a constant return with r. The problem is original given in section 8.4 of Rao and Jelvis.

Our goal is to maximize the expected utility of wealth at the final step $t = T - 1$ by dynamically allocation x_t at risky asset and the remaining $W_t - x_t$ in the risk free asset for each $t = 0, 1, \ldots, T-2$. Assume the single-time-step discount factor is γ and the utility function is CARA, which means $U(W_T) = \frac{1-e^{-\alpha W_T}}{\alpha}$.

III. Project Code Usage Instruction

You can access all the project code from https://github.com/BrightBlu/RLAssetAllocation.

You can reproduce all our experiments by specifying the experiment number. For example, to replicate Experiment 3.2.1, you can directly run:

```
python train.py --config
experiment/experiment_3_2_1/config.json
```

The results will be saved in the `results` folder. We have used a fixed random seed to ensure the reproducibility of the results.

For analyzing the experimental outcomes, you can extract the Q-Table and the Final Policy by running:

```
python extract_results.py model.npy
```

To simulate the environment and expected rewards, you can explore the `environment_simulation.ipynb` notebook.

The environment and agents have been thoroughly validated and unit-tested. You can verify this by running:

```
python -m pytest
```

Further instruction can be found in `README.md`.

IV. Mathematical Derivation

We first adopt a mathematical approach to obtain a comprehensive perspective on this problem and to develop clearer intuition regarding the form of the optimal policy. This section is structured as follows: we first derive an analytical solution for the optimal policy, analyze the conditions necessary for its existence, and then examine further properties, such as the sign (positivity or negativity) of the solution and its dependence on independent variables.

A. Value Function Approach

As the environment is given and well defined, we try to get a close-form solution first.

The problem is to maximize

$$\mathbb{E}[\gamma^{T-t}\frac{1-e^{-\alpha W_T}}{\alpha}|(t,W_t)]$$

Since γ^{T-t} and α is constant, we only need to maximize

$$\mathbb{E}[\frac{-e^{-\alpha W_T}}{\alpha}|(t,W_t)]$$

.

The correlation between two step of wealth is

$$\begin{aligned} W_{t+1} &= x_t\left(1+Y_t\right)+\left(W_t-x_t\right)\left(1+r\right) \\ &= x_t\left(Y_t-r\right)+W_t(1+r) \end{aligned}$$

The Bellman Optimality Equation in this problem is defined as

$$V_t^*(W_t)=\begin{cases}\max\limits_{x_t}(\mathbb{E}_{Y_t}[V_{t+1}^*(W_{t+1})]), & \text{For } t\leq T-2\\ \max\limits_{x_t}(\mathbb{E}_{Y_{T-1}}[\frac{-e^{-\alpha W_T}}{\alpha}]), & \text{For } t=T-1\end{cases}$$

We assume the the solution of $V_t^*(W_t)$ has a form of $-d_te^{-c_tW_{t+1}}$.

1) When $t \leq T-2$: Then we can rewrite the Bellman Optimality Equation for $t \leq T-2$ as

$$\begin{aligned} V_t^*\left(W_t\right) &= \max_{x_t}\left(\mathbb{E}_{Y_t}\left[V_{t+1}^*\left(W_{t+1}\right)\right]\right) \\ &= \max_{x_t}\left(\mathbb{E}_{Y_t}\left[-d_{t+1}\cdot e^{-c_{t+1}\cdot W_{t+1}}\right]\right) \\ &= \max_{x_t}\left(\mathbb{E}_{Y_t}\left[-d_{t+1}\cdot e^{-c_{t+1}\cdot(x_t(Y_t-r)+W_t(1+r))}\right]\right) \\ &= \max_{x_t}\Big(-d_{t+1}\cdot\Big[pe^{-c_{t+1}\cdot(x_t(a-r)+W_t(1+r))} \\ &\qquad +(1-p)e^{-c_{t+1}\cdot(x_t(b-r)+W_t(1+r))}\Big]\Big) \\ &= \max_{x_t}\Big(-d_{t+1}\cdot e^{-c_{t+1}\cdot W_t(1+r)} \\ &\qquad \Big[pe^{-c_{t+1}\cdot x_t(a-r)}+(1-p)e^{-c_{t+1}\cdot x_t(b-r)}\Big]\Big) \end{aligned}$$

So we can obtain the optimal Q-Function as

$$\begin{aligned} Q_t^*\left(W_t,x_t\right) = &-d_{t+1}\cdot e^{-c_{t+1}\cdot W_t(1+r)} \\ &\Big[pe^{-c_{t+1}\cdot x_t(a-r)}+(1-p)e^{-c_{t+1}\cdot x_t(b-r)}\Big] \end{aligned}$$

Take derivative, we can obtain

$$\begin{aligned} Q_t^{'*} = -d_{t+1}e^{-c_{t+1}(1+r)W_t}\Big(&-c_{t+1}e^{-c_{t+1}(a-r)x_t}p(a-r) \\ &-c_{t+1}e^{-c_{t+1}(b-r)x_t}(1-p)(b-r)\Big)=0 \end{aligned}$$

Solved as

$$x_t=\frac{1}{c_{t+1}(b-a)}\ln\left(-\frac{(b-r)(1-p)}{p(a-r)}\right)$$

Checked that the second derivative

$$\begin{aligned} Q_t^{''*}(W_t,x_t) = &-d_{t+1}e^{-c_{t+1}W_t(1+r)} \\ &\Big[c_{t+1}^2(a-r)^2pe^{-c_{t+1}x_t(a-r)} \\ &+c_{t+1}^2(b-r)^2(1-p)e^{-c_{t+1}x_t(b-r)}\Big] \end{aligned}$$

If d_{n+1} is positive, the second derivative is negative, vice versa.

Then the Bellman Optimality Equation can be simplified as Equation (*).

Comparing with $V_t^*(W_t)=-d_te^{-c_tW_t}$, we have

$$\begin{aligned} d_t &= d_{t+1}\left[p\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-a)}{(b-a)}}\right. \\ &\qquad \left.+(1-p)\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-b)}{(b-a)}}\right] \\ c_t &= c_{t+1}(1+r) \end{aligned}$$

2) When $t=T-1$: We can simplify the Bellman Optimality Equation as

$$\begin{aligned} V_{T-1}^*(W_{T-1}) &= \max_{x_{T-1}}\left(\mathbb{E}_{Y_{T-1}}\left[\frac{-e^{-\alpha W_T}}{\alpha}\right]\right) \\ &= \max_{x_{T-1}}\left(\mathbb{E}_{Y_{T-1}}\left[\frac{-e^{-\alpha(x_{T-1}(Y_{T-1}-r)+W_{T-1}(1+r))}}{\alpha}\right]\right) \\ &= \max_{x_{T-1}}\left(\frac{-1}{\alpha}\Big[pe^{-\alpha(x_{T-1}(a-r)+W_{T-1}(1+r))}\right. \\ &\qquad \left.+(1-p)e^{-\alpha(x_{T-1}(b-r)+W_{T-1}(1+r))}\Big]\right) \end{aligned}$$

Similarly take derivative and set derivative as 0, we can solve

$$x_{T-1}=\frac{1}{\alpha(b-a)}\ln\left(\frac{-(b-r)(1-p)}{p(a-r)}\right)$$

Then we can simplify V_{T-1}^* as

$$\begin{aligned} V_{T-1}^* = \frac{-1}{\alpha}e^{-\alpha W_{T-1}(1+r)}&\left[p\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-a)}{(b-a)}}\right. \\ &\left.+(1-p)\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-b)}{(b-a)}}\right] \end{aligned}$$

We can also have

$$\begin{aligned} d_{T-1} &= \frac{1}{\alpha}\left[p\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-a)}{(b-a)}}\right. \\ &\qquad \left.+(1-p)\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-b)}{(b-a)}}\right] \end{aligned}$$

$$c_{T-1}=\alpha(1+r)$$

$$
\begin{aligned}
V_t^*(W_t) &= -d_{t+1}e^{-c_{t+1}W_t(1+r)}\left[pe^{\frac{(a-r)}{(b-a)}\ln\left(\frac{-(b-r)(1-p)}{p(a-r)}\right)} + (1-p)e^{\frac{(b-r)}{(b-a)}\ln\left(\frac{-(b-r)(1-p)}{p(a-r)}\right)}\right] \\
&= -d_{t+1}e^{-c_{t+1}W_t(1+r)}\left[p\left(\frac{-(b-r)(1-p)}{p(a-r)}\right)^{\frac{(r-a)}{(b-a)}} + (1-p)\left(\frac{-(b-r)(1-p)}{p(a-r)}\right)^{\frac{(r-b)}{(b-a)}}\right] \\
&= -d_{t+1}\left[p\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-a)}{(b-a)}} + (1-p)\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-b)}{(b-a)}}\right]e^{-c_{t+1}W_t(1+r)} \qquad (*)
\end{aligned}
$$

Combining the results when $t \leq T-2$, we have

$$
\begin{aligned}
d_t &= \frac{1}{\alpha}\left[p\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-a)}{(b-a)}}\right. \\
&\qquad \left. +(1-p)\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-b)}{(b-a)}}\right]^{T-t} \\
c_t &= \alpha(1+r)^{T-t}
\end{aligned}
$$

And the optimal policy π^* for each step is

$$
\pi^*_{t,W_t} = x_t = \frac{1}{\alpha(1+r)^{T-t-1}(b-a)}\ln\left(\frac{(r-b)(1-p)}{p(a-r)}\right)
$$

We will discuss the effective range later.
The Optimal Value Function is

$$
\begin{aligned}
V_t^*(W_t) = -\frac{1}{\alpha}e^{-\alpha(1+r)^{T-t}W_t}&\left[p\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-a)}{(b-a)}}\right. \\
&\left. +(1-p)\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-b)}{(b-a)}}\right]^{T-t}
\end{aligned}
$$

Optimal Q-Function is

$$
\begin{aligned}
Q_t^*(W_t, x_t) = &-\frac{e^{-\alpha(1+r)^{T-t}W_t}}{\alpha} \\
&\left[pe^{-\alpha(1+r)^{T-t-1}x_t(a-r)} + (1-p)e^{-\alpha(1+r)^{T-t-1}x_t(b-r)}\right] \\
&\left[p\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-a)}{(b-a)}}\right. \\
&\qquad \left. +(1-p)\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-b)}{(b-a)}}\right]^{T-t-1}
\end{aligned}
$$

B. More insights on optimal policy function π^*

Recall that we have got the optimal policy

$$
\pi^* = x_t = \frac{1}{\alpha(1+r)^{T-t-1}(b-a)}\ln\left(\frac{(r-b)(1-p)}{p(a-r)}\right)
$$

1) Existence Conditions for Solutions: To ensure the existence of π^*, the Q-function must be convex, i.e., $Q''^*(W_t, x_t) < 0$. Therefore, we must have

$$
\begin{aligned}
d_{T-1} = \frac{1}{\alpha}&\left[p\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-a)}{(b-a)}}\right. \\
&\left. +(1-p)\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-b)}{(b-a)}}\right] > 0.
\end{aligned}
$$

Assuming $\alpha > 0$, we require

$$
\begin{aligned}
&p\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-a)}{(b-a)}} \\
&\qquad + (p-1)\left(\frac{(r-b)(1-p)}{p(a-r)}\right)^{\frac{(r-b)}{(b-a)}} > 0.
\end{aligned}
$$

Note that since $p > 0$ and $1-p > 0$, if $\frac{r-b}{r-a} > 0$, all exponents are well-defined. Under these conditions and $\alpha > 0$, we can guarantee the convexity of Q. Moreover, this condition ensures $\frac{(r-b)(1-p)}{p(a-r)} > 0$, making $\ln\frac{(r-b)(1-p)}{p(a-r)}$ valid and confirming that x_t is a legitimate solution.

If $\alpha < 0$, let us define $X = \frac{(r-b)(1-p)}{p(a-r)}$, $Y = \frac{(r-a)}{(b-a)}$, and $Z = \frac{(r-b)}{(b-a)}$. Given that $Y, Z \in \mathbb{R}^*$, the expressions $X^Y < 0$ or $X^Z < 0$ would not be properly defined. Therefore, to ensure a valid solution, we must assume $\alpha > 0$, corresponding to risk-averse behavior.

Additionally, we note that since a, b, and r represent return rates, the quantities $1+a$, $1+b$, and $1+r$ should be non-negative. For simplicity and without loss of generality, we assume they are strictly positive.

Thus, the appropriate constraints for valid solutions are $a > r > b > -1$ and $\alpha > 0$.

2) Independent Variables: The only variable in π^* is t, the time step.

We should not be surprised that the optimal policy function have nothing to do with current wealth W_t. The reason is quite obvious, we use CARA as utility function, which remains constant risk aversion during different asset levels.

3) Positivity and Negativity: As $\alpha > 0$ and $a > b$, the positivity and negativity is solely depend on

$$
\ln\frac{(r-b)(1-p)}{p(a-r)}
$$

.

Solve

$$\frac{(r-b)(1-p)}{p(a-r)} = 1$$

We get

$$p = \frac{r-b}{a-b}$$

Thus when $p > \frac{r-b}{a-b}$, the solution is strictly positive for all t. When $p < \frac{r-b}{a-b}$, the solution is strictly negative for all t. When $p = \frac{r-b}{a-b}$, the solution is all 0.

4) Edge condition: Then calculate the below, assume $b < a$ and $\alpha > 0$:

$$\lim_{b \to a^-} \frac{1}{\alpha(1+r)^{T-t-1}(b-a)} \ln\left(\frac{(r-b)(1-p)}{p(a-r)}\right) = \pm\infty$$

When $p > 0.5$, the limitation result is $+\infty$. When $p < 0.5$, the limitation result is $-\infty$. When $p = 0.5$, this limitation is not well-defined in the $\mathbb{R}$ space.

This limitation approach indicate that for some special parameter combination, the absolute value of optimal weight could be extremely large.

5) Proportionality: When $p \neq \frac{r-b}{a-b}$, we can find that w_t is a geometric sequence with ratio $(1+r)$. When $r = 0$, the weight is constant.When $r \neq 0$, $\ln|\pi^*|$ should be linear with t. We can re-write $ln|\pi^*|$ as

$$\ln|\pi^*| = kt + b'$$

where

$$k = \ln(1+r)$$

$$b' = -(T-1)\ln(1+r) + \ln\left|\frac{\ln\left(\frac{(r-b)(1-p)}{p(a-r)}\right)}{\alpha(b-a)}\right|$$

The positivity and negativity of π^* can be determined by comparing p and $\frac{r-b}{a-b}$.

C. Conclusion

Based on the discussion above, we now have a clearer understanding of the optimal policy. The optimal policy is independent of the current wealth level W_t. Its positivity or negativity is determined solely by the parameters a, b, p, and r and remains constant over time. Furthermore, by taking the logarithm, we observe that $\log|\pi^*|$ exhibits a linear relationship with respect to t.

V. Simulation Approach

We assume a set of parameter for further calculation and experiment:

$$a = 0.04$$
$$b = 0.01$$
$$p = 0.06$$
$$r = 0.02$$
$$\alpha = 0.1$$
$$T = 10$$

As $a > r > b$, $p = 0.6 > \frac{r-b}{a-b} = \frac{1}{3}$, this parameter set is valid and we expect to have a positive geometric sequence.

Then we can calculate the optimal strategy, the result is approximately

$$w_t = 306.4232 \cdot 1.02^t, t \in [0, 9]$$

By taking log,

$$\ln w_t = 0.0198t + 5.7250$$

But we are more interested about the the expected final reward. We can use Monte-Carlo method to have a approximation.

Algorithm 1 Monte-Carlo Simulation to estimate final reward under policy π

```
Initialize environment env with a, b, r, p, α
Initialize empty list result_pi.
for i = 1 to N do
    Reset environment: env.reset()
    for t = 0 to T − 1 do
        Take action according to policy: action ← π*[t]
        (state, reward, done) ← env.step(action)
        if done then
            Append final reward to list
            break
        end if
    end for
end for
Compute mean : reward ← mean(result_pi)
Compute standard deviation: σ_reward ← std(result_pi)
return reward, σ_reward
```

For this setting, we get $\mathbb{E}[R_9] = 7.923$ and $\sigma_{R_9} = 7.2038$.

VI. Agent Approach

Designing a meaningful, convergent, and correct policy is extremely challenging. In fact, achieving a fully correct solution may be beyond the scope of this analysis. Therefore, we focus on developing a practical policy approach, aiming for convergence and correctness within a reasonable range. Our objective in this section is to start with the agent-based approach from a conceptual standpoint, explore its convergence properties, and identify potential limitations and areas for further improvement.

Our basic setting of the problem is the same in the Simulation Approach. We would like to start in the simplest case. We use Q-Learning for experiments first.

The Q-Learning updates its policy by

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right]$$

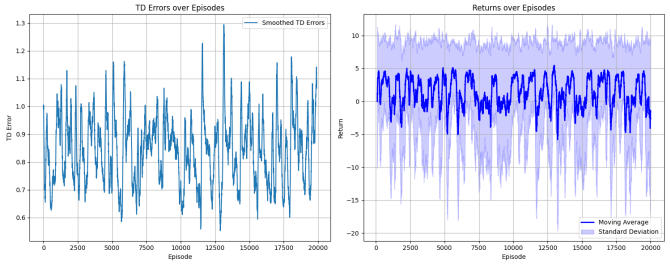

Fig. 1. Training TD Error and Reward for Experiment 1.1.1

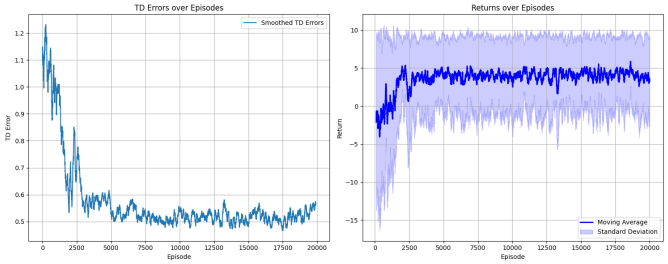

Fig. 2. Training TD Error and Reward for Experience 1.1.2 with Decayed ϵ and LR

A. *State Space Without Wealth and Action Space* $\{+, -\}$

The MDP of this case is defined as

$$\begin{aligned} \mathcal{M} &= \{\mathcal{S}, \mathcal{A}, P, R, \gamma\} \\ \mathcal{S} &= \mathbb{N}\{t \mid t \in [0, 9]\} \\ \mathcal{A} &= \{+, -\} \\ P(t+1 \mid t, \forall a \in \mathcal{A}) &= 1 \\ R(t \leq T-2) &= 0 \\ R(T-1) &= \frac{1 - e^{-\alpha W_T}}{\alpha} \\ \gamma &= 0.8 \end{aligned}$$

We begin by assuming a simplified scenario to analyze the policy. Specifically, we consider a setting in which there are only two possible actions: constantly short 300 or constantly long 300, i.e., $\mathcal{A} = \{300, -300\}$.

We choose the value 300 as it is close to the analytical optimal solution of $306.4232 \cdot 1.02^t$. Therefore, this simplification provides a reasonable approximation, facilitating a clearer analysis.

Additionally, we note that the optimal policy derived earlier is independent of the current wealth level W_t, which further justifies restricting our analysis to a constant action space.

1) Attempt with $T = 3$: We firstly start from a very simple case. We set a constant $\epsilon = 0.1$ and learning rate $= 0.1$.

The result shown in Figure 1 is very bad. This may because we have a constant learning rate and epsilon. We then try the following setting:

$$\begin{aligned} \epsilon_{\text{Start}} &= 0.5 \\ \epsilon_{\text{floor}} &= 0.001 \\ \text{Decay}_{\epsilon} &= 0.999 \\ \text{LR}_{\text{Start}} &= 0.4 \\ \text{LR}_{\text{floor}} &= 0.001 \\ \text{Decay}_{\text{LR}} &= 0.999 \end{aligned}$$

The ϵ and LR is decayed via constant ratio of Decay.

The final policy of Experiment 1.1.2 is

$$\pi(t) = \{t = 0 : +, t = 1 : +, t = 2 : +\}$$

which is the optimal policy π^*.

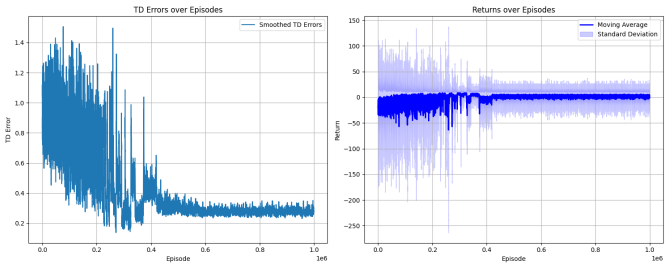

Fig. 3. Training TD Error and Reward for Experiment 1.2.1

Such experiment result in Figure 2 shows the important of ϵ and LR optimization. The decay can help us to explore more at first and have a robust result in the end. The following experiment will all be optimized on these and will not be specifically mentioned.

2) Attempt with $T = 8$: With similar setting, we have the results of the case $T = 8$ in Figure 3. The policy given by the agent is

$$\pi(t) = \{t = 2, 3 : -, \text{otherwise} : +\}$$

.

This is not the optimal policy, because it contains $-$.

However, no matter how we adjust the LR and ϵ, we just can't get the optimal policy.

We try an approach by adding an instant reward to each action with

$$\text{Reward} = \begin{cases} \text{Sigmoid}(\frac{W_t - W_{t-1}}{W_{t-1}}) - 0.5, & t = 1, \cdots, T-2 \\ \frac{1 - e^{-\alpha W_T}}{\alpha}, & t = T-1 \end{cases}$$

The Sigmoid function is defined as $S(x) = \frac{1}{1+e^x}$. We choose this because it can help us normalize any possible return into the range $(-0.5, 0.5)$, and when return is very small, our reward is closely linear.

It should be noted that such modifications may potentially alter the original intended objective of the results. Nevertheless, by incorporating this instantaneous reward, the model obtains a more explicit optimization target, thereby mitigating concerns regarding gradient stability.

The result is satisfied, as shown in Figure 4, and the final policy is

$$\pi(t) = \{\forall t, t : +\}$$

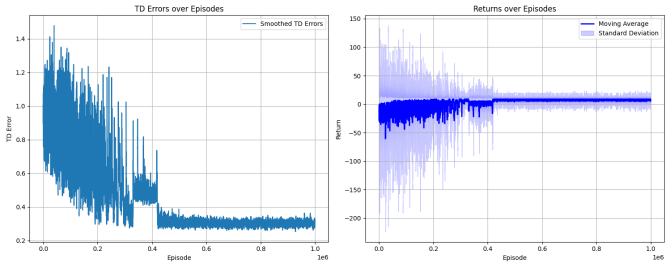

Fig. 4. Training TD Error and Reward for Experiment 1.2.2 with instant reward

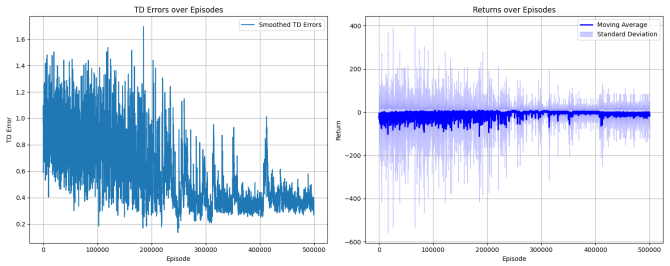

Fig. 5. Training TD Error and Reward for Experiment 1.3.1 without instant reward

which is the optimal policy.

3) Attempt with $T = 10$: The experiment of using Instant Reward in $T = 10$ further demonstrate its importance, seen in Figure 5 and Figure 6. The one with instant reward reached $\pi^* = \{\forall t, t : +\}$ and the other doesn't.

This experiment was deliberately designed rather than chosen arbitrarily. It represents the simplest possible scenario for our problem and demonstrates that a reinforcement learning paradigm is capable of learning an entirely positive or entirely negative policy. This insight is crucial for constructing the action space in subsequent experiments. Additionally, this simplified experiment serves to validate the correctness and effectiveness of our Q-learning algorithm.

B. State Space Without Wealth and Time and Action Space $\{e^N, N \in \mathbb{Z}\}$

Instead of directly searching for exact numerical values, we aim to observe whether the Q-learning agent can first approximate the correct order of magnitude and consistently maintain the same action at each step.

As previously discussed, π maintains a constant sign (positive or negative). For simplicity in this parameter configuration, we consider only the case where $\pi > 0$.

On one hand, this approach slightly increases the problem's complexity; on the other hand, since our actual action space is $\mathbb{R}$, conducting the search on an exponential scale enables comprehensive coverage of both extremely small and large values.

The MDP of this case is defined as

$$\begin{aligned}
\mathcal{M} &= \{\mathcal{S}, \mathcal{A}, P, R, \gamma\} \\
\mathcal{S} &= \{t = 0, t = 9\} \\
\mathcal{A} &= \{[\underbrace{e^N, \cdots, e^N}_{T}], N \in \mathbb{Z}\} \\
P(T-1 \mid t = 1, \forall a \in \mathcal{A}) &= 1 \\
R(t \leq T-2) &= 0 \\
R(T-1) &= \frac{1 - e^{-\alpha W_T}}{\alpha} \\
\gamma &= 0.8
\end{aligned}$$

For this question, the original optimal $\ln w_t = 0.0198t + 5.7250$. When $N = 5$, the simulation result is about 6.765. When $N = 6$, the simulation result is about 7.785. So in this case, $\pi^* = [\underbrace{e^6, \cdots, e^6}_{T}]$

Action	Q-value
0	1.23
1	1.36
2	1.69
3	2.57
4	4.27
5	6.81
6	7.81
7	-807.42
8	-1.58×10^8
9	-3.95×10^{25}
10	-7.63×10^{101}

TABLE I
Q-TABLE OF EXPERIMENT 2

We set the Action Space $\mathcal{A}$ as $N \in [0, 10]$. Table I shows the result Q-Table, which indicate the agent can correctly find the order of magnitude of the answer.

C. State Space Without Wealth and Action Space $\{e^N, N \in \mathbb{Z}\}$

We have verified that the agent can identify the correct exponential scale within a constant action space. We then aim to investigate whether the agent can discover the appropriate action space for all state spaces of t.

1) Attempt with $T = 5$: We first attempt with an easier version: $T = 5$ and $\mathcal{A} = \{e^5, e^6, e^7\}$ and no instant reward.

We can find in State $t = 4$, the policy is not optimal. This indicate the huge difficulty for a stable global optimized policy.

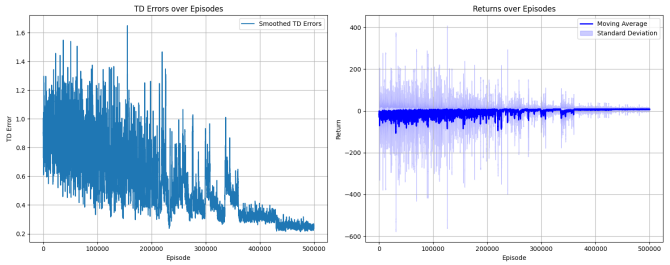

Fig. 6. Training TD Error and Reward for Experiment 1.3.2 with instant reward

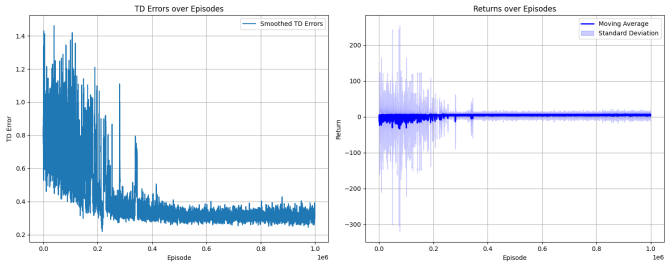

Fig. 7. Training TD Error and Reward for Experiment 3.1

	Q(s,5)	Q(s,6)	Q(s,7)
1	2.09	2.20	2.09
2	2.60	2.73	2.61
3	3.26	3.38	3.25
4	4.19	4.05	4.05
5	4.77	5.22	2.61

TABLE II
Q-TABLE WITH TRUNCATED VALUES

2) Attempt with $T = 10$: We continue to demonstrate the difficulty of this question when facing a larger action space and state space.

Firstly, we just set $T = 10$, the training results show in Figure 8. The result policy is

$$\pi(t) = \{t = 0, 4, 6, 7 : N = 5, \text{otherwise} : N = 6\}$$

Comparing with the action state $\mathcal{A} = \{e^5, e^6, e^7\}$, the result is hardly satisfying.

We then increase the action state to $\mathcal{A} = \{e^N, N \in [2, 8]_{\mathbb{Z}}\}$.

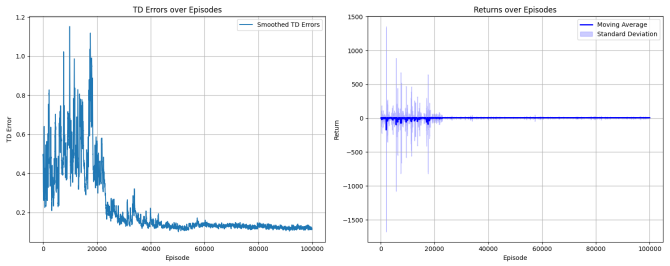

Fig. 8. Training TD Error and Reward for Experiment 3.2.1

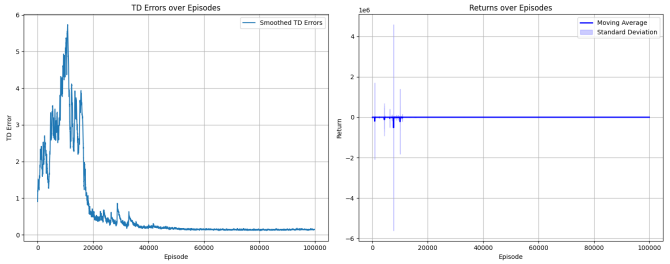

Fig. 9. Training TD Error and Reward for Experiment 3.2.1

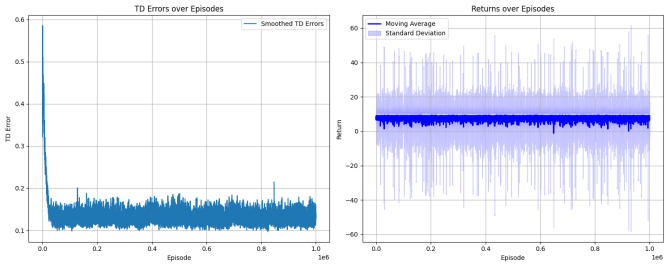

Fig. 10. Training TD Error and Reward for Experiment 4

The training results show in Figure 9 and the result policy is

$$\pi = [N \mid 5, 5, 4, 2, 5, 2, 6, 3, 6, 6]_t$$

which is far from the optimal. Although shown in TD error, the policy has been convergent, it's still very hard to get to the optimal.

D. State Space Without Wealth and Action Space $\{e^{kN+b}, N \in \mathbb{Z}\}$

Although finding the optimal magnitude in every state has proven challenging, we observe that the agent's optimization trajectory remains generally aligned with the optimal solution's direction. This leads us to investigate whether the agent can acquire the geometric sequence property characteristic of optimal strategies when provided with appropriate magnitude constraints.

We have been previously proved that

$$\ln w_t = 0.0198t + 5.7250$$

We can take $k = 0.0198, b = 5.7250$. Given $N(t) = t$, the action e^{kN+b} is the optimal policy π^*.

By constructing an Action Space $\mathcal{A} = \{e^{kN+b} \mid N \in \mathbb{Z}\}$, we can examine whether N can be fallen into $[0, 1, \cdots, T-1]$. With such operation, we can shift an exponential solution space into a linear space.

Taken $N \in [-5, 15]$ in this case, the result is very bad. The final policy is

$$\pi = [N_t \mid 8, 11, 12, 14, -2, 12, 13, 14, 15, 3, 14]_t$$

which is far away from the $\pi^* = [N_t \mid N_t = t]$.

However, after analysis the reward of the policy, we should not be surprised. Recall the optimal solution of this problem is 7.923. Using Monte-Carlo simulation, this policy has an expected return of 7.824 and a standard deviation of 10.520.

And we can continue analyze the worst case in our setting: $N_t = -5$ and $N_t = 15$, the actual action value is 280.34 and 416.55, the expected returns are 7.830 and 7.720 with standard deviation of 5.24 and 13.86.

Such setting is too close to the optimal and too far from the gradient. With such a tremendous standard deviation, it's nearly impossible to get the actual order of N by Q-Learning.

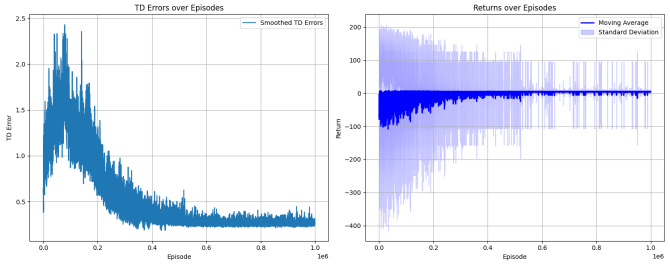

Fig. 11. Training TD Error and Reward for Experiment 5

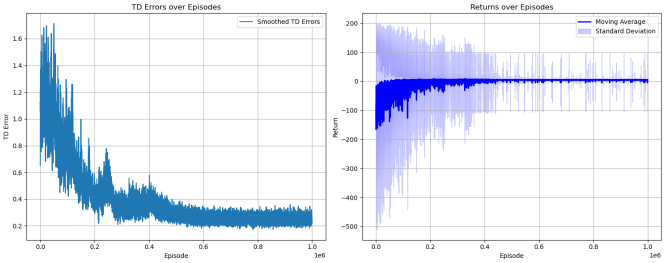

Fig. 12. Training TD Error and Reward for Experiment 6

We summarize the key findings of this phase: By providing properly calibrated ϵ and LR parameters along with immediate reward signals, the agent can successfully attain the correct constant positivity in policy. When constraining the policy to be time-invariant (constant with respect to t), the agent demonstrates capability in learning the appropriate order of magnitude for the solution. However, when actions must be applied universally across all states, the policy may deviate from optimality. Furthermore, the agent encounters difficulty in acquiring the geometric sequence property, primarily due to diminished gradient signals and elevated variance in the learning process.

E. State Space With Wealth and Action Space $\{e^N, N \in \mathbb{Z}\}$

Based on the previous discussion, we decide to just use the Action Space

$$\mathcal{A} = \{e^{N_A}, N_A \in \mathbb{Z}\}$$

for a more decisive and concentrated result.

As the action space is exponential, we can also design an exponential state space. Define

$$\mathcal{S} = \{(W_t = \pm e^{N_S}, t), N_S \in \mathbb{Z}\}$$

.

We store the state as $((\text{Sign}, \lfloor \ln |W_t| \rfloor), t)$.

In our experiment, we set $N_A \in [3, 8]_{\mathbb{Z}}$, $N_S \in [-2, 2]_{\mathbb{Z}}$.

The training results are presented in Figure 11, with the corresponding policy details documented in Table III. Although the policy does not achieve global optimality, it demonstrates convergent behavior toward π^*. Notably, the variance across different W_t values at fixed time steps t remains relatively small. This suggests the agent has successfully captured the policy's fundamental characteristic of wealth insensitivity.

F. State Space With Wealth and Action Space $\{\pm e^N, N \in \mathbb{Z}\}$

We now validate our framework through empirical testing of the original problem formulation. Employing parameter configurations consistent with Experiment 5, we introduce strategic modifications to the action space. Crucially, given the real-valued nature of actions ($x_t \in \mathbb{R}$), our methodology explicitly accounts for non-positive values in the solution space.

$$\mathcal{A} = (\text{sign}, \lfloor \ln |x_t| \rfloor) = \text{sign} \cdot e^{\ln |x_t|}$$

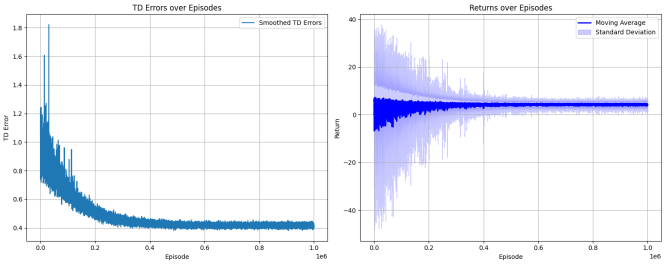

Fig. 13. Training TD Error and Reward for Experiment 7.1

The training results are presented in Figure 12, with the corresponding policy specifications detailed in Table IV. While the policy does not achieve optimality, we observe with interest that it maintains predominantly positive outputs despite the absence of immediate reward incentives in our configuration.

G. Other Reinforcement Learning Methods Approach

1) TD(0): While both TD(0) and Q-learning belong to the temporal difference (TD) learning family, they employ distinct update mechanisms. TD(0) operates as a policy-evaluation method that iteratively updates state-value estimates through the rule:

$$V(s) \leftarrow V(s) + \alpha[r + \gamma V(s') - V(s)]$$

This on-policy algorithm focuses on estimating the value function under the current policy. Our experimental results presented in Figure 13 demonstrate that TD(0) achieves comparable efficacy to Q-learning while offering two distinct advantages: superior numerical stability in extreme value regimes and more reliable convergence properties.

2) SARSA: SARSA and Q-Learning constitute distinct temporal difference approaches for estimating the optimal action-value function $Q(s, a)$. Their fundamental divergence emerges through update mechanisms:

- SARSA (on-policy):

  $$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma Q(s', a') - Q(s, a)\right]$$

  updates using the subsequent action a' selected by the current policy
- Q-Learning (off-policy):

  $$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a)\right]$$

$T-t$ \ Wealth	(1,0)	(1,1)	(1,2)	(0,-1)	(0,2)	(0,1)	(0,-2)	(1,-1)	(1,-2)	(0,0)
10	4									
9		5	4	4		7				
8	4	5	4		7		4	5		4
7	4	4	4		7	6		4	4	
6	4	5	4	4	7	5	4	4	6	4
5	4	5	4	4	7	5	4	4	4	4
4	4	5	4	4	7	4	4	4	4	4
3	4	5	4	4	7	5	4	5	5	4
2	4	5	4	4	7	4	6	6	6	4
1	6	4	6	5	7	5	4	5	5	5
0	5	7	5	5	4	7	6	5	4	6

TABLE III
POLICY TABLE OF EXPERIMENT 5

$T-t$ \ Wealth	(1,0)	(0,1)	(0,2)	(1,1)	(1,2)	(0,0)	(0,-1)	(0,-2)	(1,-1)	(1,-2)
10	(1,4)									
9	(1,4)	(1,6)	(1,7)	(1,5)	(1,5)		(1,4)	(1,4)		
8			(1,7)		(1,4)	(1,4)		(1,4)		(1,4)
7			(1,7)	(1,4)	(1,4)	(1,4)	(1,4)		(1,4)	(1,4)
6		(1,5)	(1,7)	(1,4)	(1,4)	(1,4)	(1,4)		(1,4)	(1,4)
5		(1,5)	(1,7)	(1,5)	(1,4)	(1,4)	(1,4)	(1,4)	(1,4)	(1,4)
4		(1,4)	(1,7)	(1,5)	(1,4)	(1,4)	(1,4)	(1,4)	(1,4)	(1,4)
3	(1,4)	(1,4)	(1,7)	(1,5)	(1,4)	(1,4)	(1,4)	(1,4)	(1,5)	(1,5)
2	(1,4)	(1,4)	(1,7)	(1,5)	(1,4)	(1,5)	(1,4)	(1,6)	(1,6)	(1,4)
1	(1,5)	(1,5)	(1,5)	(1,5)	(1,5)		(1,4)	(1,5)	(1,5)	(1,5)
0	(1,4)	(0,7)	(1,5)	(0,4)	(1,5)	(0,7)	(1,4)	(1,4)	(1,5)	(0,6)

TABLE IV
POLICY TABLE OF EXPERIMENT 5

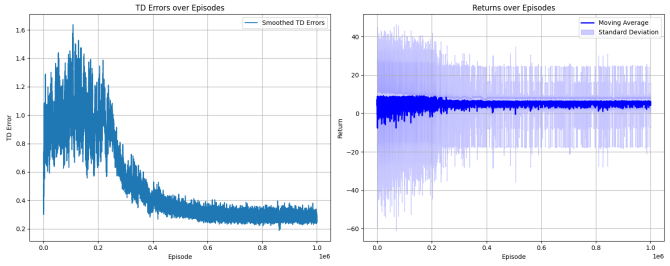

Fig. 14. Training TD Error and Reward for Experiment 7.2

directly approximates the optimal policy through maximum future reward

The experimental results presented in Figure 14 reveal SARSA's relative underperformance in this particular problem configuration, exhibiting slower convergence rates compared to both Q-Learning and TD(0) approaches. This empirical evidence aligns with theoretical expectations given the problem's inherent need for aggressive value maximization.

3) Monte-Carlo: Monte-Carlo (MC) methods constitute a class of model-free reinforcement learning techniques that estimate value functions through complete episode returns. Operating under the fundamental principle of computing empirical averages:

$$V(s) = \frac{1}{N(s)} \sum_{i=1}^{N(s)} G_t^{(i)}$$

where $G_t^{(i)}$ represents the cumulative discounted reward from episode i, these methods differ fundamentally from TD approaches by eliminating bootstrap dependencies. Our empirical evaluation, documented in Figure 15, reveals two critical limitations: inherent absence of temporal difference error metrics due to non-bootstrapping updates, and comparatively inferior convergence rates with longer training durations than TD counterparts. This performance gap originates from MC's requirement for complete episode termination before value updates.

4) DQN: Though we employed DQN as a numerical baseline, this approach yielded suboptimal performance. We provide these experimental outcomes in Figure 16 for comparative illustration.

VII. CONCLUSION

In this report, we explored the asset allocation problem using both analytical and reinforcement learning approaches.

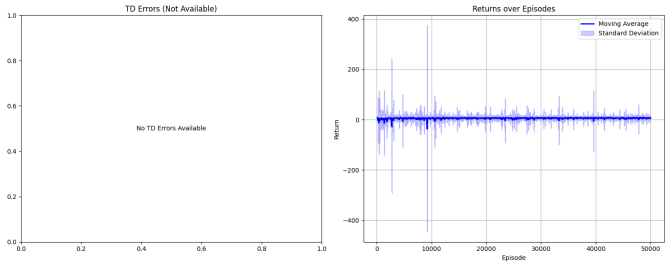

Fig. 15. Reward for Experiment 7.3, TD Error Not Suitable

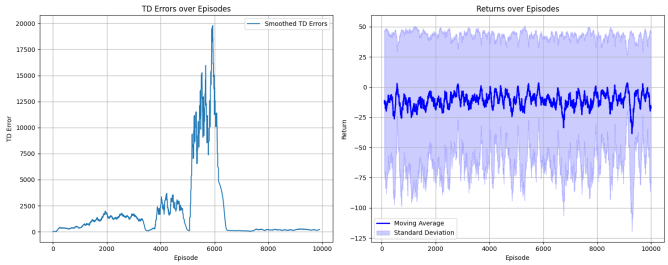

Fig. 16. Training TD Error and Reward for Experiment 7.4

We first derived the optimal policy using a mathematical formulation and analyzed its properties, including conditions for existence, proportionality, and sensitivity to independent variables. We then validated our results through Monte-Carlo simulations.

For reinforcement learning, we implemented Q-Learning and tested different state and action space representations. Our experiments demonstrated that while Q-Learning could approximate the optimal policy, convergence remained challenging in high-variance settings. We further compared Q-Learning with other reinforcement learning methods, such as TD(0), SARSA, Monte Carlo, and DQN.

Our findings highlight the potential of reinforcement learning in solving asset allocation problems while also emphasizing the importance of carefully designing the state and action spaces. Future work could explore more advanced techniques, such as function approximation and policy gradient methods, to further improve convergence and performance.