# Reinforcement Learning for Loan Pricing

Anuj Panwar

May 7, 2025

**GitHub Repository:** https://github.com/anujpanwarma2024/Summer-Project-1/tree/main

## Abstract

This report investigates reinforcement learning (RL) for pricing loans in a dynamic environment, benchmarking against the Black-Scholes-Merton (BSM) model. Traditional pricing approaches are inflexible to borrower changes, whereas RL offers adaptability through interaction-based policy learning. We model loan pricing with continuous actions sampled from a Gaussian policy and rewards reflecting pricing error. Three algorithms—REINFORCE, REINFORCE with Baseline, and TD(0)—are evaluated. TD(0) yields the most consistent and accurate results.

## 1 Introduction

Loan pricing remains a central challenge in financial modeling, with traditional methods like discounted cash flows and structural models relying on fixed borrower assumptions. These approaches fail to adapt to evolving borrower risk or economic shifts. Reinforcement learning (RL) presents a framework where an agent can learn pricing strategies through repeated interactions, updating its model based on observed reward signals.

We let the agent use borrower features to make pricing decisions and receive feedback proportional to deviations from the BSM model. Over time, this allows adaptive, data-driven pricing models capable of real-time learning.

## 2 Problem Formulation: State, Action, and Reward

We model loan pricing where each loan entry represents a state the agent observes, and the action corresponds to a pricing decision.

### State Space ($s \in \mathbb{R}^5$)

Each state is a 5-dimensional normalized vector:

- **Loan Age**: Number of months since the loan origination.
- **Borrower FICO Score**: A proxy for credit risk.
- **Loan Term (Years)**: Duration of the loan.
- **Interest Rate (%)**: Annual rate offered to borrower.
- **Qualifying DTI**: Debt-to-income ratio.

Formally,

$$s = \phi(x) = [\text{LoanAge}, \text{FICO}, \text{Term}, \text{IR}, \text{DTI}]$$

### Action Space ($a \in [0, 1]$)

Actions are scalar multipliers $a_t \sim \pi(a|s_t)$ sampled from a Gaussian policy:

$$\pi(a|s) = \mathcal{N}(\mu = \theta^\top \phi(s), \sigma^2)$$

The action predicts a synthetic price as $P_{\text{pred}} = a \cdot \text{Original Loan Amount}$. This approach allows differentiable sampling from a continuous action distribution.

### Reward Function ($r$)

The reward penalizes deviation from the BSM theoretical price:

$$r(s, a) = -\left|a \cdot \text{LoanAmount} - P_{\text{BSM}}\right|$$

This reward structure encourages accurate approximation of BSM-based pricing.

## 3 Reinforcement Learning Methods

We implement three RL algorithms for policy learning:

### 1. REINFORCE

REINFORCE uses Monte Carlo rollouts to compute the return:

$$G_t = \sum_{k=t}^{T} \gamma^{k-t} r_k$$

$$\theta \leftarrow \theta + \alpha G_t \nabla_\theta \log \pi(a_t|s_t)$$

This method estimates policy gradients from full episodes, which introduces high variance but unbiased updates.

### 2. REINFORCE with Baseline

To reduce variance, we subtract a baseline $b_t$ from returns:

$$\theta \leftarrow \theta + \alpha(G_t - b_t)\nabla_\theta \log \pi(a_t|s_t)$$

The baseline is updated using:

$$b_t = (1-\beta)b_{t-1} + \beta G_t$$

This dampens the noise in learning caused by stochastic returns.

### 3. TD(0)

TD(0) updates the value function using bootstrapping:

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

$$\theta \leftarrow \theta + \alpha \cdot \delta_t \cdot \nabla_\theta \log \pi(a_t|s_t)$$

TD(0) updates after each time step, leading to faster convergence and reduced variance compared to REINFORCE.

Each method learns a weight vector $\theta \in \mathbb{R}^5$ that maps state features to action distributions. Policy learning thus becomes a matter of refining how loan features contribute to predicted pricing weights.

# 4 Experimentation

## Environment Setup

Synthetic loan data was generated, with each loan consisting of normalized features and an original amount. At each timestep of an episode:

- A loan entry was randomly sampled and treated as state $s_t$.
- The policy generated a mean $\mu_t = \theta^\top \phi(s_t)$, then sampled an action $a_t \sim \mathcal{N}(\mu_t, \sigma^2)$.
- The predicted price was calculated, and reward $r_t$ was determined by comparing to the BSM price.

## Hyperparameters

- **Learning Rate** ($\alpha = 0.005$): Allows smooth but effective gradient updates.
- **Discount Factor** ($\gamma = 0.95$): Promotes long-term reward optimization.
- **Action Variance** ($\sigma = 0.2$): Injects controlled exploration into the Gaussian policy.
- **Baseline Smoothing** ($\beta = 0.1$): Balances adaptivity and stability in baseline tracking.

### Training Procedure

Training was conducted over 200 episodes with 4 timesteps per episode. Each algorithm used its respective update logic:

- REINFORCE and REINFORCE+Baseline updated $\theta$ after each episode.
- TD(0) updated $\theta$ at every step using temporal-difference errors.

Each policy maintained a single $\theta \in \mathbb{R}^5$, and all actions remained in the bounded interval $[0, 1]$.

## 5 Results and Analysis

The average returns (i.e., negative pricing errors) over 200 episodes are summarized below:

| **Method** | **Mean Return** | **Performance** |
|---|---|---|
| REINFORCE | $-85{,}667.80$ | High variance |
| REINFORCE + Baseline | $-83{,}826.56$ | Noisy convergence |
| TD(0) | $-75{,}579.04$ | Best overall |

**REINFORCE:** Achieved learning but suffered from unstable updates due to full episode dependency. Fluctuating returns hindered convergence.

**REINFORCE + Baseline:** Some variance reduction was observed, but poor baseline estimates occasionally led to suboptimal gradient steps, slowing down learning.

**TD(0):** Most stable performance with consistent improvement over episodes. Real-time updates enabled quicker adjustment to reward signals and better generalization.

## 6 Conclusion and Limitations

Reinforcement learning demonstrates potential for adaptive pricing in finance. Among the evaluated methods, TD(0) most effectively minimized deviation from theoretical prices with fast and stable convergence.

**Limitations:**

- Evaluation was limited to synthetic data.
- Scalar action limits pricing granularity.
- No sequential temporal dependencies were modeled.

**Future Work:** Exploring real-world data, broader action models, and sequential deep RL (e.g., LSTM, Transformer) architectures can further improve practical applicability.

## References


[1] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press.

[2] Black, F., & Scholes, M. (1973). The Pricing of Options and Corporate Liabilities. *Journal of Political Economy*, 81(3), 637–654.

[3] Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4), 229–256.

[4] Moody, J., Wu, L., Liao, Y., & Saffell, M. (2001). Performance functions and reinforcement learning for trading systems and portfolios. *Journal of Forecasting*, 17(5-6), 441–470.