Reinforcement Learning for Pricing American Options

Shamimeh Heydari

Thesis submitted to the University of Ottawa
in partial Fulfillment of the requirements for the
Master of Science, Mathematics

Department of Mathematics and Statistics
Faculty of Science
University of Ottawa

François-Michel Boire, Maia Fraser

Abstract

This thesis develops a martingale-based reinforcement-learning (RL) framework for pricing European and American options. For each learning mode, a shared European option price surface is learned from risk-neutral paths and then used as the baseline for an American early-exercise correction. Offline learning uses complete simulated paths, whereas online learning uses one-step martingale increments. The American option premium-learning stage uses a penalized, entropy-regularized two-action stopping formulation, and prices are evaluated through the deterministic stopping rule obtained after training. Under Black–Scholes dynamics, maximum American execution-price errors are 2.40% offline and 2.97% online. Under Merton jump–diffusion dynamics, a direct transfer of the Black–Scholes implementation did not adequately capture early exercise. Improving state-space and exercise-date coverage and reorganizing the updates while retaining the two-stage formulation yields offline and online mean errors of 1.31% and 1.09%, with maxima of 2.28% and 2.09%, respectively. Both methods recover meaningful early-exercise behaviour.

Acknowledgements

I am deeply grateful to my supervisors, Professor François-Michel Boire and Professor Maia Fraser, for their exceptional guidance, encouragement, and support throughout my studies. Their generosity, patience, and confidence in my work have meant a great deal to me, both academically and personally. I am especially thankful for the time, care, and thoughtful advice they have given me at every stage of this thesis. Their support provided me with the stability and motivation I needed to complete this work.

I would also like to thank the University of Ottawa and the Interdisciplinary Mathematics and Artificial Intelligence Program, Inter-Math-AI (IMA), for their support. The opportunities and resources I received during my program allowed me to focus on my research and to develop this project.

To my husband, I am deeply thankful for standing beside me with love, patience, and understanding throughout this journey. His support gave me strength in difficult moments, and his belief in me helped me continue when the path felt uncertain. I am grateful for his presence, encouragement, and kindness every step of the way.

Finally, I thank my family, whose love and encouragement have stayed with me despite the distance and the years we have spent apart.

This thesis would not have been possible without the support, patience, and kindness of all these people.

Author Contributions

This thesis was prepared as a single-author Master's thesis. Unless otherwise stated, the work presented in this thesis was carried out by me under the supervision of Professor François-Michel Boire and Professor Maia Fraser.

The thesis builds on existing work on martingale-based learning in continuous time and on the penalty and exploratory-control formulation of optimal stopping. I do not claim to have invented these general theoretical frameworks. My contribution is their adaptation, implementation, numerical investigation, and interpretation in the setting of European and American put option pricing.

Some implementation choices in the European policy-evaluation stage were inspired by author-provided code associated with the martingale-learning framework of Jia and Zhou (2022). That code served as a reference for learning a terminal-conditioned European value function from simulated Black–Scholes paths using offline and online procedures. The implementation used in this thesis was, however, extensively adapted and extended. In particular, I reformulated the experiment for put options, reorganized the code for the thesis setting, added benchmark comparisons for prices and Greeks, and integrated the European learner into the later American premium-learning stage.

A central contribution of this thesis is the two-stage learned-baseline approach. In the first stage, I train a European value approximation from simulated risk-neutral paths. In the second stage, I use this learned European value as a frozen baseline for the American premium-learning problem. This construction allows the American network to focus on the early-exercise correction and produces a deterministic stopping rule after training.

I also generalized the European and American learning problems across a panel of contracts and model parameters. Instead of training separate networks for each contract and parameter specification, I trained shared contract- and parameter-conditioned networks. For each learning mode, one European network and one American premium network are used across the contract family. This turns the method from a single-contract experiment into a reusable pricing representation.

In addition to the Black–Scholes benchmark experiments, I extended the numerical study to the Merton jump–diffusion setting. I trained parameter-conditioned networks

over a curated pool of moderate jump-diffusion specifications and evaluated each learning mode at a representative moderate-jump specification. This extension tests the two-stage martingale-based structure under finite-activity jumps after adapting state-space and exercise-date coverage and reorganizing the optimization updates for the Merton setting.

I was responsible for implementing the numerical experiments, preparing the benchmark comparisons, producing the tables, analyzing the pricing and Greek errors, and reporting the runtime and memory measurements. I also implemented the Black–Scholes comparison between the pre-trained and closed-form European option price baselines in the American stage.

The numerical experiments were run by me using computational resources on Narval provided through the Digital Research Alliance of Canada and its regional partners. I was responsible for preparing and submitting the computational jobs, collecting the outputs, and incorporating the computational results into the thesis.

My supervisors provided guidance on the research direction, mathematical formulation, numerical design, and presentation of the thesis. They reviewed drafts and provided comments throughout the project. The final implementation, numerical experiments, written presentation, and interpretation of results are my responsibility.

Contents

Notation Summary

Table 1: Notation summary

Symbol	**Meaning**
Probability, market variables, and contracts	
Ω	sample space
$\mathcal{F}$	sigma-algebra on Ω
$(\mathcal{F}_t)_{0\le t\le T}$	filtration representing the information available up to time t
$\mathbb{Q}$	risk-neutral probability measure
$\mathbb{E}^{\mathbb{Q}}$	expectation under the risk-neutral measure $\mathbb{Q}$
$W = (W_t)_{0\le t\le T}$	one-dimensional Brownian motion under $\mathbb{Q}$
t	current time
T	maturity time
$T - t$	time remaining to maturity
$t_n,\ t_\ell$	grid points used in the discrete-time implementation
$\Delta t,\ \Delta t_n$	time-step size. Here $\Delta t_n = t_{n+1} - t_n$, with uniform Δt in the numerical experiments
s	deterministic stock-price state variable
s_0	initial stock price in the theoretical SDE
S_0	initial stock price used in the numerical experiments
$S = (S_t)_{0\le t\le T}$	stock-price process
K	strike price
r	risk-free interest rate
q	continuous dividend yield
σ	diffusion volatility parameter
$g_K(s) = (K - s)^+$	payoff function of a put option with strike K
$\mathbf{1}_{\{\cdot\}}$	indicator function
ATM, ITM, OTM	at-the-money, in-the-money, and out-of-the-money contracts

Continued on next page

Symbol	Meaning (continued)
$\mathcal{L}$	Black–Scholes generator, $$(\mathcal{L}f)(t,s) = f_t(t,s) + (r-q)sf_s(t,s) + \frac{1}{2}\sigma^2 s^2 f_{ss}(t,s)$$ for smooth f
Merton jump–diffusion dynamics	
S_{t-}	stock price immediately before time t, used in the jump SDE
$N^J(dt, dy)$	Poisson random measure for jumps with mark y
η_J	jump arrival intensity in the Merton jump–diffusion model
$F_Y(dy)$	distribution of the log-jump size Y
$Y \sim N(\mu_J, \delta_J^2)$	normal log-jump size in the Merton model
μ_J	mean of the log-jump size
δ_J	standard deviation of the log-jump size
e^Y	multiplicative jump factor in the stock price
$\kappa = \mathbb{E}[e^Y - 1]$	compensator of the relative jump size, $$\kappa = \exp\left(\mu_J + \frac{1}{2}\delta_J^2\right) - 1$$
$N_{t,T}^J$	number of jumps over $(t, T]$, where $N_{t,T}^J \sim \text{Poisson}(\eta_J(T-t))$
$\mathcal{L}_M$	Merton jump–diffusion generator, $$(\mathcal{L}_M f)(t,s) = f_t(t,s) + (r - q - \eta_J\kappa)sf_s(t,s) + \frac{1}{2}\sigma^2 s^2 f_{ss}(t,s) + \eta_J \int_{\mathbb{R}} [f(t, se^y) - f(t,s)]\, F_Y(dy)$$ for smooth f
European option values and European option price learning	
$V_E(t,s)$	European put value under Black–Scholes dynamics
$V_E^M(t,s)$	European put value under Merton jump–diffusion dynamics
$V_{E,n}^M(t,s)$	conditional Merton European put value given $N_{t,T}^J = n$

Continued on next page

Symbol	Meaning (continued)
$d_1(t,s)$, $d_2(t,s)$	Black–Scholes quantities used in the European put formula
$d^M_{1,n}(t,s)$, $d^M_{2,n}(t,s)$	Merton quantities used in the conditional European put formula
Φ	cumulative distribution function of the standard normal law
ϕ	density of the standard normal law
$\Delta_E, \Gamma_E, \mathrm{Vega}_E, \Theta_E$	European benchmark delta, gamma, vega, and theta
θ	trainable parameter vector of the European option price network
$\mathcal{P}$	parameter space for the European option price-network parameters
$\mathcal{N}^E_\theta$	scalar-output neural network used in the European option price approximation
$V_{E,\theta}(t,s;T,K,\sigma)$	parametric approximation of the European put value under Black–Scholes dynamics
$V^M_{E,\theta}(t,s;T,K,\sigma,\eta_J,\mu_J,\delta_J)$	parametric approximation of the European put value under Merton dynamics
$\widehat{V}_E(t,s;T,K,\sigma)$	pre-trained European option price approximation held fixed as the Black–Scholes European option price baseline
$\widehat{V}^M_E(t,s;T,K,\sigma,\eta_J,\mu_J,\delta_J)$	pre-trained European option price approximation held fixed as the Merton European option price baseline
$Y_t = e^{-r(T-t)} g_K(S_T)$	discounted terminal payoff used in the European offline regression loss
$M^E_t = e^{-rt} V_E(t,S_t)$	true discounted European option value process
$M^{E,\theta}_t = e^{-rt} V_{E,\theta}(t,S_t)$	approximate discounted European option value process
$\mathcal{L}^{\mathrm{ML}}_E(\theta)$	theoretical European offline martingale loss
$\mathcal{L}^{\mathrm{reg}}_E(\theta)$	implemented European offline regression loss
$\widehat{\mathcal{L}}^{\mathrm{reg}}_E(\theta)$	time-discretized implemented European regression loss
$\mathrm{DMSVE}_E(\theta)$	discounted mean-square value error for the European option price approximation
$\mathrm{MSVE}_E(\theta)$	mean-square value error for the implemented European regression problem

Continued on next page

Symbol	Meaning (continued)
$\delta^{\theta}_{E,n}$	one-step European martingale or TD-type residual in the online method
α_n	learning rate in the European online update
$\ell^{\mathrm{on}}_{E,n}(\theta)$	local online surrogate objective for the European CTD(0)-type update
$\mathrm{sg}(\cdot)$	stop-gradient operator
American option values and stopping formulation	
τ	stopping time
$\mathcal{T}_{t,T}$	set of admissible stopping times taking values between t and T
$V_A(t,s)$	American put value under Black–Scholes dynamics
$V_A^M(t,s)$	American put value under Merton jump–diffusion dynamics
$\mathcal{C}$	continuation region, $\{(t,s) : V_A(t,s) > g_K(s)\}$
$\mathcal{S}$	stopping region, $\{(t,s) : V_A(t,s) = g_K(s)\}$
Γ	penalty factor in the American option premium-learning formulation
λ	entropy temperature or exploration weight
$u \in \{0,1\}$	binary action. The action $u = 0$ corresponds to continuation, and $u = 1$ corresponds to the penalized stopping action
π_v, π_n	Bernoulli stopping probabilities at future time v and grid point t_n, conditioned on the relevant state, contract, and asset-price model parameters
$H(\pi) =$ $\pi \log \pi + (1-\pi)\log(1-\pi)$	negative Bernoulli entropy. The quantity $-H(\pi)$ is the Bernoulli entropy
American option premium learning with a pre-trained European option price baseline	
$\widetilde{V}_A(t,s;T,K,\sigma)$	American option value measured relative to the pre-trained European option price baseline, $\widetilde{V}_A = V_A - \widehat{V}_E$

Continued on next page

Symbol	Meaning (continued)
$\widehat{g}(t,s;T,K,\sigma)$	baseline-adjusted exercise obstacle, $$\widehat{g}(t,s;T,K,\sigma) = g_K(s) - \widehat{V}_E(t,s;T,K,\sigma)$$
$\widehat{g}^M(t,s;T,K,\sigma,\eta_J,\mu_J,\delta_J)$	Merton baseline-adjusted exercise obstacle, $$\widehat{g}^M = g_K - \widehat{V}_E^M$$
$\mathcal{R}_{\widehat{E}}(t,s;T,K,\sigma)$	residual of the pre-trained European option price baseline under Black–Scholes dynamics, $$(\mathcal{L}\widehat{V}_E)(t,s;T,K,\sigma) - r\widehat{V}_E(t,s;T,K,\sigma)$$
$U_A(t,s;T,K,\sigma)$	American option correction learned relative to the pre-trained European option price baseline
$U_{A,\Gamma}(t,s;T,K,\sigma)$	penalized approximation of the American option correction
$U^{\lambda}_{A,\Gamma}(t,s)$	optimized penalized and entropy-regularized American option correction
θ_A	trainable parameter vector of the American option premium network
$\mathcal{N}^A_{\theta_A}$	scalar-output neural network used in the American option correction approximation
$U_{A,\theta_A}(t,s;T,K,\sigma)$	neural approximation of the American option correction under Black–Scholes dynamics
$U^M_{A,\theta_A}(t,s;T,K,\sigma,\eta_J,\mu_J,\delta_J)$	neural approximation of the American option correction under Merton dynamics
$z_{\theta_A}(t,s)$	raw scalar output of the final Merton American option premium network used to parameterize the decision gap
c_{gap}	positive scale in the final Merton decision-gap parameterization, with $c_{\text{gap}} = \lambda_0/\Gamma = 0.0025$

Continued on next page

Symbol	Meaning (continued)
λ_0	initial entropy temperature in the final Merton implementation, with $\lambda_0 = 0.025$
$\widehat{V}_A(t,s)$	reconstructed learned American option value, $$\widehat{V}_A = \widehat{V}_E + U_{A,\theta_A}$$
$\widehat{V}_A^M(t,s)$	reconstructed learned American option value under Merton dynamics, $$\widehat{V}_A^M = \widehat{V}_E^M + U_{A,\theta_A}^M$$
$\rho^\pi(v,S_v)$	fixed-policy running term in the American option premium-learning formulation, $$\rho^\pi(v,S_v) = \Gamma\widehat{g}(v,S_v)\pi_v - \lambda H(\pi_v)$$ with $\widehat{g}$ replaced by $\widehat{g}^M$ under Merton dynamics
$R_{t,v}^\pi$	penalty-induced weighting from starting time t to future time v, $$\frac{d}{dv}R_{t,v}^\pi = -\Gamma\pi_v R_{t,v}^\pi, \qquad R_{t,t}^\pi = 1$$
R_n	discrete penalty-induced weighting used in the numerical implementations, $$R_{n+1} = R_n\left(1 - \Gamma\pi_n\Delta t\right), \qquad R_0 = 1$$
$J_{\widehat{E}}^\pi(t,s)$	fixed-policy American option correction under Bernoulli policy π
$\mathcal{H}_{\widehat{E}}(t,s,\pi;w)$	Hamiltonian used in the learned-baseline Bernoulli policy-improvement step
$\pi_{\widehat{E}}^*(t,s;w)$	Bernoulli policy-improvement rule for a value argument w

Continued on next page

Symbol	Meaning (continued)
$M^{\pi}_{t,v}$	reward-adjusted martingale for the correct fixed-policy American option correction
$M^{\pi,\theta_A}_{t,v}$	approximate reward-adjusted process obtained by replacing $J^{\pi}_{\widehat{E}}$ by U_{A,θ_A}
$\mathcal{L}^{\mathrm{off}}_{A,\pi}(\theta_A)$	continuous-time offline American option martingale loss for a fixed policy π
$\widehat{\mathcal{L}}^{\mathrm{off}}_{A,\pi}(\theta_A)$	time-discretized offline American option martingale loss
$\mathrm{WMSVE}_{A,\pi}(\theta_A)$	weighted mean-square value error for the fixed-policy American option premium-evaluation problem
$G^{\pi}_{\ell}(\theta_A)$	discrete backward residual used in the offline American option loss
$\delta^{\theta_A}_{\ell}$	one-step American option martingale or TD-type residual in the online method
δ^{M}_{n}	complete one-step Merton American option residual, including the exact one-step financial discount and the frozen European option price-network martingale-defect correction
α_{ℓ}	learning rate in the American online update
$\ell^{\mathrm{on}}_{\ell}(\theta_A)$	local online surrogate objective for the American option update
$u_{\theta_A}(t,s)$	deterministic execution rule derived from the learned American option premium network
τ_{θ_A}	learned stopping time induced by u_{θ_A} on the exercise grid
$P_{\mathrm{stop}}, P_{\mathrm{stop},0}$	stopping-based American option price, with $P_{\mathrm{stop},0}$ denoting its initial-time value
Benchmarking, errors, and numerical evaluation	
$V^{E,*}_0$	European option benchmark price at the initial time
$\widehat{V}^{E}_0$	learned European option price at the initial time
$V^{A,\mathrm{ref}}_0$	American option reference price at the initial time
$V_{\mathrm{Bench},0}$	American option benchmark price used in the baseline-comparison tables

Continued on next page

Symbol	Meaning (continued)
$V_{\text{exec,learned},0}$	stopping-based American option price using the pre-trained European option price baseline
$V_{\text{exec,closed},0}$	stopping-based American option price using the closed-form European option price baseline
RelErr	relative pricing error, $$\text{RelErr}(P, P_{\text{bench}}) = \frac{\lvert P - P_{\text{bench}}\rvert}{P_{\text{bench}}}.$$ The numerical tables report this quantity as a percentage after multiplication by 100
$\Delta_A, \Gamma_A, \text{Vega}_A, \Theta_A$	American option benchmark delta, gamma, vega, and theta
$\widehat{\Delta}_A, \widehat{\Gamma}_A, \widehat{\text{Vega}}_A, \widehat{\Theta}_A$	American option Greeks computed by finite differences of the stopping-based execution-price surface
Price L^2	empirical mean squared price error over the 61-point initial-stock-price grid $[0.5S_0, 1.5S_0]$
Delta L^2, Gamma L^2, Vega L^2, Theta L^2	empirical mean squared Greek errors over the same grid
Offline	learning mode based on full-return martingale residuals constructed from complete simulated paths
Online	learning mode based on one-step martingale residuals and CTD(0)-type objectives. In the final Merton implementation, several one-step contributions may be batched before an optimizer update

List of Tables

1

Introduction

Derivative pricing provides a natural meeting point between probability, partial differential equations, numerical analysis, and sequential decision-making. In an arbitrage-free market, the value of a derivative can be represented as the expectation, under a risk-neutral measure, of its discounted future cash flows. This probabilistic viewpoint also leads, in Markovian models, to dynamic pricing equations. For European options, where exercise occurs only at maturity, this connection produces a fixed-terminal-time valuation problem.

The Black–Scholes model (Black and Scholes, 1973; Merton, 1973) is the classical example, giving formulas for European prices and Greeks.

American options introduce an additional layer of difficulty. Their payoff is not determined only by the terminal asset value, because the holder may exercise before maturity. Pricing therefore becomes an optimal stopping problem: one must determine both the value of the contract and the region in which exercise is optimal (Shreve, 2004). Analytically, this leads to a variational inequality and a free boundary separating continuation from stopping (Peskir and Shiryaev, 2006; Friedman, 1982). Numerically, the challenge is not only to estimate a price, but also to recover a reliable exercise rule.

Classical methods for American options include tree schemes, finite-difference methods, penalty methods, and simulation-based procedures (Glasserman, 2004). These methods are powerful and remain essential benchmarks. Their structure, however, is often tailored to a particular model, grid, or numerical solver. This thesis instead asks how the probabilistic structure of option pricing can be used to train value approximations directly from simulated risk-neutral paths, while retaining a common two-stage learned-baseline structure and using shared contract- and model-parameter-conditioned networks within each asset-price experiment.

The central tool is the martingale characterization of value processes. When the correct option value is evaluated along a risk-neutral path and discounted appropriately, the resulting process satisfies a martingale identity (Shreve, 2004). This observation turns option valuation into a learning problem: rather than fitting directly to closed-form formu-

las, the networks are trained using martingale-loss or martingale-orthogonality objectives constructed from simulated paths. Following Jia and Zhou (2022), the thesis develops two complementary learning procedures. The offline procedure uses future-return objectives along complete simulated paths, whereas the online procedure uses one-step temporal-difference (TD)-type martingale residuals between consecutive grid points.

This thesis addresses the American option pricing problem through a hierarchical learned-baseline premium construction. This construction builds on the penalty and exploratory control formulation of optimal stopping developed by Dai et al. (2025), together with the martingale policy-evaluation framework of Jia and Zhou (2022). In the American put option experiment of Dai et al. (2025), the early-exercise premium is learned relative to the closed-form European option value available under Black–Scholes dynamics. In this thesis, the European option price baseline is instead learned from simulated risk-neutral paths. The terminal-conditioned parameterization explicitly embeds the contractual payoff g_K, so the neural-network component learns the European option's time-value correction $\widehat{V}_E - g_K$. After training, the shared European option price surface is held fixed and used as the European option price baseline during the American option pricing stage. Therefore, the American option premium network does not require a closed-form European option value as its baseline during training.

The learned-baseline premium construction consists of two hierarchical stages. In the first stage, the terminal-conditioned European option value model represents the European option value as the contractual payoff plus a learned time-value premium, or correction. In the second stage, the American option premium model learns an additional premium

relative to the pre-trained European option price baseline. When the European option price baseline is exact, the premium learned in the second stage coincides with the usual early-exercise premium. Both learned premiums vanish at maturity, so the two neural networks do not need to relearn the terminal payoff. Within each learning mode and asset-price experiment, one European option price network and one American option premium network are shared across the contract panel. The Merton networks are also conditioned on the jump–diffusion parameters. Separate network pairs are trained for the offline and online procedures. The final American option price is obtained by executing the deterministic stopping rule induced by the learned American option premium.

The numerical study considers two asset-price dynamics. The Black–Scholes diffusion setting is used as the controlled benchmark case, where the underlying paths are continuous. The second setting uses Merton jump–diffusion dynamics, under which the stock price includes finite-activity jumps (Merton, 1976). In the Merton experiment, parameter-conditioned networks are trained over a curated pool of moderate jump-diffusion specifications and evaluated at one representative specification. The two-stage construction is retained, while state-space and exercise-date coverage and optimizer updates are adapted to the jump–diffusion setting. The adapted procedures recover meaningful early-exercise regions and produce American stopping-based prices close to the dynamic-programming benchmarks, although the learned European option price surface is less accurate than under Black–Scholes dynamics.

Modern option pricing begins with the arbitrage-based framework of Black and Scholes (1973) and Merton (1973). In this framework, no-arbitrage valuation is linked to

risk-neutral expectations. In Markovian diffusion settings, European claims can also be characterized by linear parabolic pricing equations, whereas American contracts lead to optimal stopping problems and variational inequalities. The Black–Scholes formula for vanilla European options remains a fundamental benchmark for prices and sensitivities. Extensions such as Merton's jump–diffusion model enlarge the asset-price dynamics by allowing discontinuous asset returns (Merton, 1976). Other extensions, such as stochastic-volatility models and generalized autoregressive conditional heteroskedasticity (GARCH) models, further relax the constant-volatility structure of the Black–Scholes model (Heston, 1993; Duan, 1995). Recent comparisons of American-option exercise algorithms have also considered price models based on geometric Brownian motion (GBM), GARCH, and exponential GARCH (EGARCH), reinforcing the relevance of richer volatility dynamics for future extensions (Kor, Bratvold, and Hong, 2024).

American option pricing is more involved because an exercise policy must be determined in addition to the option value. The resulting optimal stopping problem can be characterized through the Snell envelope, dynamic programming, and variational inequalities (Peskir and Shiryaev, 2006; Friedman, 1982). The continuation and stopping regions are not given in advance; they must be identified, or approximated, as part of the solution. This free-boundary structure explains why American options require numerical methods even in settings where the corresponding European option is available in closed form.

A large numerical literature addresses these valuation problems. Tree methods provide standard lattice approximations (Cox, Ross, and Rubinstein, 1979), while finite-difference methods provide grid-based PDE approximations. Penalty methods replace the obstacle

constraint by a penalized equation that can be solved numerically (Forsyth and Vetzal, 2002). Simulation-based methods are also central in computational finance. Monte Carlo methods were introduced into option valuation by Boyle (1977), and regression-based simulation methods for American-style options were developed in works such as Carriere (1996), Tsitsiklis and Van Roy (2001), and Longstaff and Schwartz (2001). These approaches are especially useful when flexibility across asset-price dynamics and payoff structures is important (Glasserman, 2004). A recent overview of American option models and algorithms, including both classical and learning-based methods, is given by Bloch (2023). The simulation of diffusion and jump–diffusion paths is supported by the broader theory of numerical methods for stochastic differential equations (SDE) (Kloeden and Platen, 1992; Øksendal, 2003; Karatzas and Shreve, 1991).

A related line of work treats American option exercise explicitly as a sequential decision-making problem. In this view, the state typically contains time and the underlying asset price, while the action set consists of the two decisions: exercise or continue. Least-squares policy iteration (LSPI), introduced by Lagoudakis and Parr (2004), provides an approximate policy-iteration method based on least-squares value-function approximation. This LSPI viewpoint was applied to American-option exercise-policy learning by Y. Li, Szepesvári, and Schuurmans (2009), where the goal is to learn exercise policies directly from simulated paths. More recently, N. Li (2022) proposed an iterative reinforcement-learning algorithm for American option pricing, based on successively approximating the expected discounted payoff of stopping rules and improving them toward optimality. The recent study by Kor, Bratvold, and Hong (2024) compares binomial, least-squares Monte

Carlo, and reinforcement-learning approaches for American put valuation, with particular emphasis on the exercise decisions and the evolution of learned stopping policies during training. These works help connect the classical regression-based American-option literature with the reinforcement-learning language of policy evaluation, policy improvement, and learned exercise rules.

RL provides a useful language for value-function approximation, policy evaluation, policy improvement, and exploration (Sutton and Barto, 2018). In this thesis, the most central parts are policy evaluation and value-function estimation, because the European option price baseline and the fixed-policy American option premium are both learned through value approximations. Classical TD learning updates value estimates from successive observations (Sutton and Barto, 2018). In continuous-time stochastic environments, however, TD methods require additional care.

The martingale approach of Jia and Zhou (2022), which is central to this work, formulates continuous-time policy evaluation through martingale identities. In that framework, a correct value function makes an associated process a martingale. This leads to two learning principles: an offline martingale loss and an online martingale-orthogonality condition. The same work also explains why a mean-square TD objective is not generally appropriate for value-function estimation in stochastic continuous time, since it is driven by quadratic variation rather than by mean-square value-function error.

This martingale perspective is particularly natural for option pricing. Under the risk-neutral measure $\mathbb{Q}$, arbitrage-free prices are expressed as conditional expectations of dis-

counted future cash flows. Consequently, when the correct value function is evaluated along a risk-neutral path and discounted, the resulting process is a martingale. For European options, this gives a fixed-maturity evaluation problem with a known terminal payoff. For American options, the stopping decision adds a control component; after a stopping policy is fixed, the same martingale policy-evaluation logic can be applied to the corresponding fixed-policy value.

The starting point for the learning approach is TD learning, introduced by Sutton (1988), in which value estimates are updated using the difference between successive predictions. Jia and Zhou (2022) extend this idea to continuous-time policy evaluation by replacing the discrete-time prediction-error viewpoint with a martingale characterization of the value process. This is the framework that justifies the offline martingale loss and the online martingale-orthogonality update used in this thesis.

The optimal-stopping framework of Dai et al. (2025) then adapts this continuous-time martingale-learning viewpoint to American-style stopping problems. In that work, the American variational inequality is approximated by a penalty formulation, rewritten as a two-action control problem, and relaxed through Bernoulli randomization with entropy regularization. This improves the basic TD/policy-evaluation framework for the American option pricing problem by adding the missing stopping component: policy improvement now updates a randomized stop-or-continue rule, while martingale-based policy evaluation estimates the value of that fixed rule. This construction provides the main methodological bridge used in this thesis for learning the American early-exercise premium and the associated stopping policy.

The relation between the present thesis and the LSPI/RL literature above is mainly through the learned stopping policy. Unlike LSPI-style discrete-time policy-iteration methods, the present approach uses a continuous-time martingale policy-evaluation framework. Unlike the American option experiment of Dai et al. (2025), which uses the closed-form European Black–Scholes price as a baseline, this thesis first learns the European value from simulated risk-neutral paths and then freezes it as the baseline for American early-exercise premium learning. Thus the method keeps the stop–continue interpretation of American exercise, while shifting the learning problem toward a reusable martingale-trained European option price baseline and a learned American option premium.

More broadly, learning-based methods have become important in numerical approaches to high-dimensional PDEs (Han, Jentzen, and E, 2018), stochastic control (Huré et al., 2021), and optimal stopping problems (Becker, Cheridito, and Jentzen, 2019; Becker, Cheridito, Jentzen, and Welti, 2021). The present thesis fits into this literature by studying a martingale-based learning framework for European and American options, with emphasis on value learning, early-exercise premium learning, deterministic execution after exploration, and robustness under both diffusion and jump–diffusion dynamics.

The scope of the thesis is therefore deliberately focused. The main contracts considered are European and American put options. The learning framework is developed under Black–Scholes GBM dynamics and then tested under a moderate Merton jump–diffusion extension. The numerical experiments compare prices, Greeks, and computational costs for offline and online learning procedures. For American options, the primary learned object is not only a value approximation, but also the deterministic stopping rule obtained after

training.

The remainder of the thesis is organized as follows. Chapter 2 introduces the option-pricing framework, including European and American contracts, risk-neutral valuation, the Black–Scholes pricing equation, the American variational inequality, and the Merton jump–diffusion extension. Chapter 3 develops the offline and online martingale-based learning procedures, including the pre-trained European baseline, the American premium formulation, and the deterministic execution rule. Chapter 4 reports the numerical results under Black–Scholes and moderate Merton jump–diffusion dynamics, including pricing errors, Greek errors, and computational costs. The thesis concludes with a summary of findings, limitations, and directions for future work.

2

Option Pricing

Option pricing is a central problem in mathematical finance. In an arbitrage-free market, the value of a derivative contract can be represented under a risk-neutral measure, that is, an equivalent probability measure under which the expected rate of return of the underlying asset, after accounting for dividends, is the risk-free rate. Under this measure, derivative prices are expressed as discounted expectations of future payoffs. Their precise form then depends on two fundamental ingredients: the stochastic evolution of the under-

lying asset and the contractual rule governing exercise. The distinction between European and American options is therefore essential. A European option may be exercised only at maturity, so its valuation is a fixed-maturity pricing problem. An American option may be exercised at any admissible time up to maturity, so its valuation is an optimal stopping problem. This difference in exercise rights leads to two closely related but mathematically distinct pricing frameworks.

To formulate the pricing problems precisely, we begin with the contractual structure of the option. The key ingredients are the payoff specification and the exercise rule. These determine the cash flow delivered by the contract and, in turn, the mathematical form of the valuation problem.

2.1 European and American Option Contracts

An option is a derivative contract written on an underlying asset and specified by a strike price $K > 0$, a maturity date $T > 0$, and an exercise rule. We represent the contractual payoff by a function

$$g_K : (0, \infty) \to [0, \infty),$$

where the subscript K indicates that the payoff may depend on the strike.

Throughout this chapter, we assume that g_K is Borel measurable, nonnegative, and of at most polynomial growth in the asset price. In particular, there exist constants $C > 0$

and $m \geq 0$, where m is a fixed finite real exponent, such that

$$0 \leq g_K(s) \leq C(1 + s^m), \qquad s > 0,$$

which is a natural regularity condition for the valuation framework used later. Under the GBM model, it ensures that $g_K(S_T)$ is square-integrable, since the stock price has finite moments of every order on finite time intervals.

A European option can be exercised only at the fixed maturity date T. The exercise time is therefore predetermined by the contract, and no exercise decision is made after the contract is initiated.

An American option is defined by the same contractual data, but the holder may exercise at any admissible time between the current date and maturity. Such an exercise time must be chosen using only the information available up to the time of exercise. Conceptually, this is the role of a stopping time: it is a random exercise time whose occurrence can be determined from the information observed so far.

The distinction between the two exercise features is therefore precise. A European option is a fixed-maturity claim, while an American option also includes the right to choose an exercise time before maturity. This additional flexibility is what turns American option pricing into an optimal stopping problem.

With the contractual distinction now in place, we next formulate both contracts in a common arbitrage-free market model and define their values under the risk-neutral measure.

2.2 Risk-Neutral Model and Valuation of European and American Options

The contractual description becomes a pricing problem only after the dynamics of the underlying asset are specified. Assuming the absence of arbitrage, we define the corresponding option values under the risk-neutral measure. Consider a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{0\le t\le T}, \mathbb{Q})$, where $\mathbb{Q}$ is the risk-neutral measure and $W = (W_t)_{0\le t\le T}$ is a one-dimensional Brownian motion under $\mathbb{Q}$.

Under $\mathbb{Q}$, the GBM process $S = (S_t)_{0\le t\le T}$ follows the stochastic differential equation (SDE)

$$dS_t = (r - q)S_t\,dt + \sigma S_t\,dW_t, \qquad S_0 = s_0 > 0, \tag{2.1}$$

where $r \ge 0$ is the risk-free rate, $q \ge 0$ is the continuous dividend yield, and $\sigma > 0$ is the volatility. This is the Black–Scholes model (Black and Scholes, 1973) under the risk-neutral measure with continuous dividend yield.

Using the payoff function g_K introduced in Section 2.1, the arbitrage-free value of the European option at time t, when the current asset price is $S_t = s$, is

$$V_E(t, s) = \mathbb{E}^{\mathbb{Q}}\left[e^{-r(T-t)}g_K(S_T) \mid S_t = s\right], \qquad (t, s) \in [0, T] \times (0, \infty). \tag{2.2}$$

Here $\mathbb{E}^{\mathbb{Q}}$ denotes expectation with respect to the risk-neutral probability measure $\mathbb{Q}$. Thus the European price is the conditional risk-neutral expectation of the discounted terminal

payoff.

For the American contract, the valuation problem is formulated over stopping times. A random variable $\tau : \Omega \to [0, T]$ is called an $(\mathcal{F}_u)_{0\leq u\leq T}$-stopping time if

$$\{\tau \leq u\} \in \mathcal{F}_u, \qquad u \in [0, T].$$

This means that the decision to exercise by time u can be made using only the information available up to that time.

For each $t \in [0, T]$, let $\mathcal{T}_{t,T}$ denote the set of $(\mathcal{F}_u)_{0\leq u\leq T}$-stopping times taking values in $[t, T]$. Thus, $[t, T]$ is the set of possible exercise dates, while $\mathcal{T}_{t,T}$ is the collection of admissible stopping times whose values lie in this interval.

Under the chosen risk-neutral measure $\mathbb{Q}$, the value of the American option is

$$V_A(t, s) := \sup_{\tau\in\mathcal{T}_{t,T}} \mathbb{E}^{\mathbb{Q}} \left[e^{-r(\tau-t)} g_K(S_\tau) \mid S_t = s\right], \qquad (t, s) \in [0, T] \times (0, \infty). \tag{2.3}$$

The deterministic stopping time $\tau \equiv T$, corresponding to the European exercise rule, belongs to $\mathcal{T}_{t,T}$. Therefore, Equation (2.3) implies

$$V_A(t, s) \geq \mathbb{E}^{\mathbb{Q}} \left[e^{-r(T-t)} g_K(S_T) \mid S_t = s\right] = V_E(t, s), \qquad (t, s) \in [0, T] \times (0, \infty). \tag{2.4}$$

This inequality shows that the right of early exercise cannot reduce the value of the con-

tract. At maturity, both contracts coincide with the payoff:

$$V_E(T,s) = V_A(T,s) = g_K(s), \qquad s > 0. \tag{2.5}$$

Thus, the European and American option values are equal at maturity.

The valuation formulas above define the option prices in probabilistic terms. It is also useful to express the same problems in dynamic form. In the Black–Scholes setting, the resulting equations describe how the option value evolves over time before maturity and make precise the distinction between pure valuation and optimal stopping.

2.3 Dynamic Valuation

We now present the differential equations associated with the two pricing problems. Under the usual regularity assumptions, the European value is characterized by a linear parabolic pricing equation, whereas the American value is characterized by a variational inequality. The latter reflects the fact that, at each time–state point, the holder may either continue holding the contract or exercise immediately.

For a sufficiently smooth function $f = f(t,s)$, define the infinitesimal generator by

$$(\mathcal{L}f)(t,s) = \frac{\partial f}{\partial t}(t,s) + (r-q)s\frac{\partial f}{\partial s}(t,s) + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 f}{\partial s^2}(t,s). \tag{2.6}$$

This operator describes the local time evolution of a smooth function along the risk-neutral

stock-price dynamics.

Under the Black–Scholes model, the European value V_E satisfies

$$\begin{cases} (\mathcal{L}V_E)(t,s) - rV_E(t,s) = 0, & (t,s) \in [0,T) \times (0,\infty), \\ V_E(T,s) = g_K(s), & s > 0. \end{cases} \tag{2.7}$$

The equation follows by applying Itô's formula to $V_E(t, S_t)$ and then discounting: the drift of $e^{-rt}V_E(t, S_t)$ is $e^{-rt}((\mathcal{L}V_E)(t, S_t) - rV_E(t, S_t))$, since the discounted European value is a martingale under $\mathbb{Q}$, this drift must be zero.

The American value V_A satisfies the variational inequality

$$\begin{cases} \max\{(\mathcal{L}V_A)(t,s) - rV_A(t,s),\, g_K(s) - V_A(t,s)\} = 0, & (t,s) \in [0,T) \times (0,\infty), \\ V_A(T,s) = g_K(s), & s > 0. \end{cases} \tag{2.8}$$

This equation combines two requirements. The term $(\mathcal{L}V_A)(t,s) - rV_A(t,s)$, corresponds to continuation, while the term $g_K(s) - V_A(t,s)$, compares immediate exercise with continuation. In particular, the variational inequality implies that $V_A(t,s) \geq g_K(s)$ throughout the domain, so the American value always dominates the immediate-exercise payoff.

This leads to the continuation and stopping regions

$$\mathcal{C} := \{(t,s) \in [0,T) \times (0,\infty) : V_A(t,s) > g_K(s)\},$$

and

$$\mathcal{S} := \{(t, s) \in [0, T) \times (0, \infty) : V_A(t, s) = g_K(s)\}.$$

On the continuation region, the American value satisfies the same linear pricing equation as a held claim,

$$(\mathcal{L}V_A)(t, s) - rV_A(t, s) = 0 \qquad \text{for } (t, s) \in \mathcal{C}.$$

On the stopping region, immediate exercise is optimal and the value is equal to the exercise payoff:

$$V_A(t, s) = g_K(s) \qquad \text{for } (t, s) \in \mathcal{S}.$$

The interface between these two regions is the free boundary, also called the early-exercise boundary. Along this boundary, the holder is indifferent between continuing and exercising.

The contrast between (2.7) and (2.8) is the main analytical difference between European and American option pricing in this setting. The European contract is described by a linear terminal-value problem, while the American contract is described by a free-boundary problem in which the stopping region is part of the solution itself.

We now specialize the payoff and record the closed-form European benchmark available in the Black–Scholes model.

2.4 Numerical Benchmarks

This section discusses the model and option settings used in the rest of the thesis and presents the benchmark quantities used in the numerical study.

We begin by distinguishing between call and put payoffs. A call option gives the holder the right to buy the underlying asset at the strike price K, while a put option gives the holder the right to sell the underlying asset at that strike. The standard vanilla payoff functions, g_K, for call and put options are $(s-K)^+$,$(K-s)^+$, respectively. The call payoff increases when the asset price rises above the strike, while the put payoff increases when the asset price falls below the strike.

Put-call symmetry relates the corresponding put and call valuation problems by interchanging the spot and strike variables and by swapping the interest rate and dividend yield (McDonald and Schroder, 1998). Thus, a pricing method developed for puts can be translated to the corresponding call problem through this symmetry. For this reason, the remainder of this thesis focuses on put options, and the payoff function is $g_K(s) = (K-s)^+$ for $s > 0$.

Under the Black–Scholes model (2.1), the European put has a closed-form valuation formula (Black and Scholes, 1973; Merton, 1973; Shreve, 2004). This formula also gives explicit benchmark Greeks. These quantities are used in the numerical study as reference values, not as training targets.

For $0 \le t < T$, define

$$d_1(t,s) := \frac{\log(s/K) + \left(r - q + \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}, \qquad d_2(t,s) := d_1(t,s) - \sigma\sqrt{T-t}, \quad (2.9)$$

which combine log-moneyness, time to maturity, drift adjustment, and volatility scale.

Let Φ denote the cumulative distribution function (CDF) of the standard normal law. Define its density by

$$\phi(z) := \frac{1}{\sqrt{2\pi}} e^{-z^2/2}, \qquad z \in \mathbb{R}.$$

This notation will be used in both the price formula and the benchmark Greek formulas.

The Black–Scholes price of the European put is

$$V_E(t,s) = Ke^{-r(T-t)}\Phi(-d_2(t,s)) - se^{-q(T-t)}\Phi(-d_1(t,s)), \qquad (t,s) \in [0,T) \times (0,\infty), \quad (2.10)$$

which is the closed-form European value associated with the put payoff $g_K(s) = (K-s)^+$.

The benchmark Greeks used in the numerical study are obtained by differentiating (2.10). To keep the main text focused on the valuation framework, the explicit formulas for delta, gamma, vega, and theta are collected in Appendix A.

2.5 Merton Jump–Diffusion Dynamics

The Black–Scholes model assumes that stock-price paths are continuous, and provides a useful benchmark, but its continuous-path assumption does not capture sudden stock-price movements. To test whether the two-stage martingale-based learning framework remains effective beyond the continuous-path Black–Scholes diffusion setting, the numerical study also considers the Merton jump–diffusion model (Merton, 1976). The Merton jump–diffusion model is used in this thesis as a controlled robustness benchmark for the learning framework rather than as an exhaustive model of stock-price jump risk.

The Merton jump–diffusion model is a natural first extension of the Black–Scholes asset-price dynamics in this thesis. It introduces finite-activity jumps while retaining a one-dimensional Markov state represented by the stock price, direct simulation of risk-neutral stock-price paths, and a tractable Poisson-mixture representation for European put option values. These properties make it possible to change the risk-neutral asset-price dynamics, infinitesimal generator, and option-pricing benchmark calculations while retaining the underlying two-stage mathematical formulation and deterministic stopping rule. However, the Merton training procedure is refined by improving state-space and exercise-date coverage, conditioning the networks on the jump–diffusion parameters, reorganizing the optimization updates, and accounting for the martingale defect of the frozen learned European option price baseline. The experiment therefore evaluates the framework under finite-activity jumps after these model-specific numerical refinements.

The numerical experiments in this thesis use Merton's Gaussian log-jump-size specification. Alternative finite-activity jump–diffusion models use other log-jump-size distributions, including the double-exponential distribution of Kou (2002) and the mixed-exponential distribution of Cai and Kou (2011). Broader extensions of the asset-price dynamics include stochastic-volatility jump models such as Bates (1996) and infinite-activity Lévy models such as the CGMY model (Carr et al., 2002). The numerical experiments in this thesis do not test these alternative jump-size distributions or broader asset-price models.

Accordingly, throughout the Merton experiment, the stock price is specified under the risk-neutral measure by

$$\frac{dS_t}{S_{t-}} = (r - q - \eta_J \kappa)\, dt + \sigma\, dW_t + \int_{\mathbb{R}} (e^y - 1)\, N^J(dt, dy), \tag{2.11}$$

where $N^J(dt, dy)$ is a Poisson random measure with intensity

$$\eta_J F_Y(dy)\, dt.$$

Here $\eta_J > 0$ is the jump arrival intensity, F_Y is the distribution of the log-jump size Y, and in the Merton model

$$Y \sim N(\mu_J, \delta_J^2).$$

The compensator of the relative jump size is

$$\kappa := \mathbb{E}[e^Y - 1] = \exp\left(\mu_J + \frac{1}{2}\delta_J^2\right) - 1. \tag{2.12}$$

The drift correction $-\eta_J\kappa$ ensures that the stock has risk-neutral expected growth rate $r - q$.

For a sufficiently smooth function $f = f(t, s)$, the corresponding Merton generator is

$$(\mathcal{L}_M f)(t,s) = \frac{\partial f}{\partial t}(t,s) + (r-q-\eta_J\kappa)s\frac{\partial f}{\partial s}(t,s) + \frac{1}{2}\sigma^2 s^2 \frac{\partial^2 f}{\partial s^2}(t,s) + \eta_J \int_{\mathbb{R}} [f(t,se^y) - f(t,s)] F_Y(dy). \tag{2.13}$$

Thus the European pricing equation and the American variational inequality have the same formal structure as in the diffusion case, but with the Black–Scholes generator $\mathcal{L}$ replaced by the jump–diffusion generator $\mathcal{L}_M$. The equations therefore become partial integro-differential equations because of the nonlocal jump term.

For European options, the Merton model also admits a closed-form series representation. This formula is used as the European benchmark in the numerical study. To obtain it, one conditions on the number of jumps over $[t, T]$, as in the standard Merton jump–diffusion derivation (Merton, 1976; Glasserman, 2004).

Conditionally on $S_t = s$, the explicit solution of (2.11) gives

$$\log S_T = \log s + \left(r - q - \eta_J\kappa - \frac{1}{2}\sigma^2\right)(T-t) + \sigma(W_T - W_t) + \sum_{i=1}^{N_{t,T}^J} Y_i,$$

where

$$N_{t,T}^J \sim \text{Poisson}(\eta_J(T-t)),$$

and $(Y_i)_{i\geq 1}$ are i.i.d. copies of Y, independent of $N_{t,T}^J$ and of the Brownian motion.

Conditional on $N_{t,T}^J = n$, the jump sum is normally distributed with mean $n\mu_J$ and variance $n\delta_J^2$. Hence $\log S_T \mid N_{t,T}^J = n$ is normal with variance

$$\sigma^2(T-t) + n\delta_J^2.$$

For $t < T$ and $n = 0, 1, 2, \ldots$, define

$$d_{2,n}^M(t,s) := \frac{\log(s/K) + \left(r - q - \eta_J\kappa - \frac{1}{2}\sigma^2\right)(T-t) + n\mu_J}{\sqrt{\sigma^2(T-t) + n\delta_J^2}}, \tag{2.14}$$

and

$$d_{1,n}^M(t,s) := d_{2,n}^M(t,s) + \sqrt{\sigma^2(T-t) + n\delta_J^2}.$$

For a fixed number of jumps n, the conditional European put value is

$$\begin{aligned} V_{E,n}^M(t,s) :=& \mathbb{E}^{\mathbb{Q}}\left[e^{-r(T-t)}(K - S_T)^+ \;\middle|\; S_t = s,\ N_{t,T}^J = n\right] = Ke^{-r(T-t)}\Phi(-d_{2,n}^M(t,s)) \\ &- s\exp\left(-q(T-t) - \eta_J\kappa(T-t) + n\left(\mu_J + \frac{1}{2}\delta_J^2\right)\right)\Phi(-d_{1,n}^M(t,s)), \end{aligned} \tag{2.15}$$

which is obtained from the conditional lognormal distribution of S_T given $N_{t,T}^J = n$.

Averaging the conditional value (2.15) over the Poisson distribution of $N^J_{t,T}$ gives

$$V^M_E(t,s) = \sum_{n=0}^{\infty} e^{-\eta_J(T-t)} \frac{(\eta_J(T-t))^n}{n!} V^M_{E,n}(t,s). \tag{2.16}$$

In computation, this infinite series is truncated after sufficiently many terms.

In the numerical study, benchmark prices are computed differently for European and American options. European benchmark prices are computed directly from the available closed-form formulas. American options do not admit comparable closed-form benchmark formulas in these settings. Their benchmark prices are therefore computed numerically on the exercise grid using a dynamic-programming Bermudan approximation, following the class of methods used for American-style option pricing in Ben-Ameur, Breton, and L'Ecuyer (2002) and the related stochastic dynamic-programming benchmark framework of Teto (2025).

3

Methodology

This chapter presents the learning methodology used to price American options through a learned European baseline and an early-exercise premium. The main object of the thesis is the American optimal stopping problem. The European construction is included for exposition and for baseline construction. The main pricing problem is the American optimal-stopping problem, which is treated through an American premium-learning component. In this component, the learned European value is held fixed and the network

learns the early-exercise correction that determines the final stopping rule.

The American premium-learning component is inspired by the penalty and exploratory-control formulation of optimal stopping in Dai et al. (2025). In their American option experiment, the authors work under the Black–Scholes diffusion model and use the closed-form European put value as the baseline for learning the early-exercise premium. Thus the learning problem is shifted from the full American value to the premium relative to the European value.

A contribution of this thesis is to replace that closed-form European baseline by a learned European approximation (Jia and Zhou, 2022). The European value is trained from simulated risk-neutral paths and then used as a frozen baseline for the American premium-learning problem. This removes the available analytic European formula from the training baseline and allows the same learned-baseline structure to be tested under both geometric Brownian motion and Merton jump–diffusion dynamics. Both models admit analytic European benchmark formulas, which are used only for evaluation in the main learned-baseline experiments. The experiments therefore test whether training can proceed without using those formulas. They do not test a model for which no analytic European pricing formula exists.

The methodology combines two martingale-based learning modes. The offline mode uses complete simulated paths and trains value approximations through full-path martingale-loss or payoff-regression objectives. The online mode uses martingale orthogonality and leads to TD-type semi-gradient updates. Both modes are rooted in the continuous-time

martingale policy-evaluation framework of Jia and Zhou (2022), where the value function is characterized through the martingality of an associated discounted process.

The chapter is organized around the two learning modes. Section 3.1 develops the offline procedure. It first records how the European baseline $\widehat{V}_E$ is learned and then derives the corresponding offline martingale loss for the American premium relative to this baseline. Section 3.2 presents the online procedure. It begins with the European martingale-orthogonality update and then gives the analogous one-step update for the American premium. In both sections, the European construction is included only to explain how the baseline used by the American method is obtained.

Before presenting the offline and online algorithms, we clarify one piece of policy-evaluation terminology used below. In the continuous-time martingale-learning framework of Jia and Zhou (2022), a value function may include a terminal payoff and an accumulated running reward. In the notation of this thesis, this generic form can be written as

$$V(t,x) = \mathbb{E}\left[\int_t^T \rho(u, X_u)\, du + g(X_T) \;\middle|\; X_t = x\right], \tag{3.1}$$

where $\rho(u, X_u)$ is the instantaneous reward rate, or running reward, and $g(X_T)$ is the terminal payoff. Thus the accumulated running reward over $[t, T]$ is the time integral of ρ. In the European option problem, there is no running reward before maturity, and the only payoff is the terminal payoff $g_K(S_T)$. In the American premium-learning problem, a running reward appears after a randomized Bernoulli stopping policy has been fixed. The specific running reward used in the American premium-learning problem is introduced

when the offline American loss function is derived below.

3.1 Offline Learning

This section develops the offline learning procedure. Although the valuation problems are formulated in continuous time, the implemented losses are computed on simulated paths observed on a grid $0 = t_0 < t_1 < \cdots < t_N = T$. Here, offline means that the loss for one update is constructed after the simulated path has been generated over the relevant time interval. The update may therefore use information from later grid points. For the European baseline, this means using the terminal payoff $g_K(S_T)$. For the American premium-learning problem, after a Bernoulli stopping policy has been fixed, this means using the accumulated future values of the corresponding running reward.

The section first specifies the terminal-conditioned European baseline approximation and the offline loss used to train it. It then introduces the learned-baseline transformation of the American problem, the penalized exploratory premium equation, and the offline martingale loss used to train the American premium network.

The first object needed by the learned-baseline method is an approximation of the European value function. This approximation is learned before the American premium-learning problem is trained and is then kept fixed. We introduce it here because the learned European value serves as the baseline in the American premium formulation.

To approximate the European put value, we use a parametric family $V_{E,\theta}(t, s; T, K, \sigma)$,

where θ denotes the trainable parameters. We write $\theta \in \mathcal{P} \subseteq \mathbb{R}^p$, where p is the total number of trainable parameters and $\mathcal{P}$ is the parameter space. In the implementation, θ contains the weight matrices, bias vectors, and the trainable batch-normalization parameters.

Inspired by the terminal-conditioned form used in the martingale-learning framework of Jia and Zhou (2022), we write the approximation as

$$V_{E,\theta}(t, s; T, K, \sigma) = g_K(s) + (T - t)\mathcal{N}_\theta^E(t, s; T, K, \sigma), \tag{3.2}$$

where $\mathcal{N}_\theta^E$ is a scalar-output neural network. The notation $\mathcal{N}_\theta^E(t, s; T, K, \sigma)$ indicates that the approximation is conditioned on the current state (t, s), the contract variables (T, K), and the model parameter σ. The exact feature vector used in the implementation is reported in the numerical results section.

This form decomposes the current European option value into its exercise value $g_K(s)$ and a learned time-value correction. The factor $T - t$ forces the time-value correction to vanish at maturity. Therefore,

$$V_{E,\theta}(T, s; T, K, \sigma) = g_K(s), \qquad s > 0,$$

so the terminal payoff condition is imposed by construction.

The offline loss function for the European baseline is based on the martingale property of the discounted European value process under $\mathbb{Q}$ (Shreve, 2004). For the true European

value, $M_t^E := e^{-rt}V_E(t, S_t)$, for all $0 \leq t \leq T$, is a $\mathcal{F}_t$-measurable $\mathbb{Q}$-martingale, and at maturity $M_T^E = e^{-rT}g_K(S_T)$. This follows from applying Itô's formula to $V_E(t, S_t)$, discounting by e^{-rt}, and using the European pricing equation (2.7).

For a parametric approximation $V_{E,\theta}$, define the approximate discounted process

$$M_t^{E,\theta} := e^{-rt}V_{E,\theta}(t, S_t). \tag{3.3}$$

Following the martingale-loss idea of Jia and Zhou (2022), one may train $V_{E,\theta}$ by comparing the approximate discounted process $M_t^{E,\theta}$ with the terminal martingale value M_T^E. Integrating this squared discrepancy over the life of the option gives the theoretical offline martingale loss

$$\mathcal{L}_E^{\mathrm{ML}}(\theta) := \frac{1}{2}\mathbb{E}^{\mathbb{Q}}\left[\int_0^T \left|e^{-rT}g_K(S_T) - e^{-rt}V_{E,\theta}(t, S_t)\right|^2 dt\right]. \tag{3.4}$$

This loss measures, along simulated paths, the discrepancy between the discounted terminal payoff and the discounted approximate value process over the life of the option.

In the implementation, we use the closely related payoff-regression form at the level of the undiscounted value. For each time t, define

$$Y_t := e^{-r(T-t)}g_K(S_T). \tag{3.5}$$

This random variable is the terminal payoff discounted back to the current time t, satisfying $V_E(t, S_t) = \mathbb{E}^{\mathbb{Q}}\left[Y_t \mid \mathcal{F}_t\right]$.

Thus the implemented offline European loss is

$$\mathcal{L}_E^{\mathrm{reg}}(\theta) := \frac{1}{2}\mathbb{E}^{\mathbb{Q}}\left[\int_0^T |Y_t - V_{E,\theta}(t, S_t)|^2 \, dt\right]. \tag{3.6}$$

The conditional-expectation argument showing why this regression target is appropriate, and how it relates to the martingale-loss objective, is given in Appendix B.1.

On the time grid $0 = t_0 < t_1 < \cdots < t_N = T$, equation 3.6 is approximated by

$$\widehat{\mathcal{L}}_E^{\mathrm{reg}}(\theta) = \frac{1}{2}\mathbb{E}^{\mathbb{Q}}\left[\sum_{n=0}^{N-1} \left|e^{-r(T-t_n)} g_K(S_T) - V_{E,\theta}(t_n, S_{t_n})\right|^2 \Delta t_n\right]. \tag{3.7}$$

This implemented regression loss admits the same projection-type decomposition as the theoretical martingale loss.

In practice, the expectation is replaced by the empirical average over simulated risk-neutral paths. After this loss is minimized, the trained European approximation is denoted by $\widehat{V}_E$ and is kept fixed for the American premium-learning problem.

We now use the pre-trained European approximation to formulate the American premium-learning problem. Let $\widehat{V}_E(t, s; T, K, \sigma)$ denote the pre-trained European baseline. This network is held fixed during the American premium-learning step, so its parameters are not updated.

The American option value is represented relative to this pre-trained European baseline:

$$\widetilde{V}_A(t, s; T, K, \sigma) := V_A(t, s; T, K, \sigma) - \widehat{V}_E(t, s; T, K, \sigma). \tag{3.8}$$

If the pre-trained European approximation were exact, then $\widetilde{V}_A$ would coincide with the true early-exercise premium. In the implemented method, it is interpreted as the American premium relative to the pre-trained European baseline.

The American exercise constraint $V_A(t,s) \geq g_K(s)$ becomes a constraint on this premium. Indeed, substituting $V_A = \widehat{V}_E + \widetilde{V}_A$ gives

$$\widetilde{V}_A(t,s;T,K,\sigma) \geq g_K(s) - \widehat{V}_E(t,s;T,K,\sigma).$$

We therefore define the baseline-adjusted exercise obstacle

$$\widehat{g}(t,s;T,K,\sigma) := g_K(s) - \widehat{V}_E(t,s;T,K,\sigma). \tag{3.9}$$

This quantity is the immediate-exercise payoff measured relative to the pre-trained European baseline. Since $\widehat{V}_E - g_K$ is the learned European time-value correction, $\widehat{g}$ is equivalently the negative of that correction. Indeed it is the obstacle that the American premium must dominate.

We denote by U_A the premium function actually learned in the American component. Equivalently, the learned American value is reconstructed as

$$\widehat{V}_A(t,s;T,K,\sigma) = \widehat{V}_E(t,s;T,K,\sigma) + U_A(t,s;T,K,\sigma).$$

Thus U_A represents the American correction relative to the pre-trained European baseline.

Both the European and American option values coincide with the payoff at maturity. Since the European approximation satisfies the terminal condition by construction, $\widehat{V}_E(T, s; T, K, \sigma) = g_K(s)$, we have $\widehat{g}(T, s; T, K, \sigma) = 0$. Accordingly, the premium relative to the pre-trained European baseline satisfies the zero terminal condition $U_A(T, s; T, K, \sigma) = 0$. This zero terminal condition is useful numerically because the American learner does not need to reproduce the nonsmooth terminal payoff. Instead, it learns only the correction associated with the early-exercise right.

Because the European baseline is pre-trained rather than exact, subtracting $\widehat{V}_E$ from the American variational inequality would introduce the residual

$$\mathcal{R}_{\widehat{E}}(t, s; T, K, \sigma) := (\mathcal{L}\widehat{V}_E)(t, s; T, K, \sigma) - r\widehat{V}_E(t, s; T, K, \sigma).$$

This residual measures how far the pre-trained European baseline is from satisfying the European pricing equation. In the implementation, this residual is not included explicitly. Instead, we learn U_A relative to the fixed baseline $\widehat{V}_E$. The implemented premium problem is therefore

$$\begin{cases} \max\{(\mathcal{L}U_A)(t, s) - rU_A(t, s),\, \widehat{g}(t, s) - U_A(t, s)\} = 0, & (t, s) \in [0, T) \times (0, \infty), \\ U_A(T, s) = 0, & s > 0. \end{cases} \tag{3.10}$$

The first term corresponds to continuation of the premium, while the second term enforces the baseline-adjusted exercise obstacle.

For computation, we do not solve (3.10) directly. Following the penalty formulation in Dai et al. (2025), the obstacle constraint $U_A(t,s) \geq \widehat{g}(t,s)$ is approximated by the penalized equation

$$\begin{cases} (\mathcal{L}U_{A,\Gamma})(t,s) - rU_{A,\Gamma}(t,s) + \Gamma(\widehat{g}(t,s) - U_{A,\Gamma}(t,s))^+ = 0, & (t,s) \in [0,T) \times (0,\infty), \\ U_{A,\Gamma}(T,s) = 0, & s > 0. \end{cases} \tag{3.11}$$

The penalty term acts only when the premium approximation falls below the exercise obstacle. It is therefore a penalty approximation of the stopping constraint, not a regularization term on the neural-network parameters.

Since $x^+ = \max_{u \in \{0,1\}} ux$, the penalized equation can be rewritten as a two-action control problem:

$$(\mathcal{L}U_{A,\Gamma})(t,s) - rU_{A,\Gamma}(t,s) + \max_{u \in \{0,1\}} \{-\Gamma u U_{A,\Gamma}(t,s) + \Gamma u \widehat{g}(t,s)\} = 0. \tag{3.12}$$

This reformulation turns the penalized stopping problem into a two-action control problem. The action $u = 0$ corresponds to continuation, while $u = 1$ corresponds to the penalized stopping action.

During training, the stopping decision is randomized through a Bernoulli policy. At a future time $v \in [t,T]$, the stopping probability is a state-dependent quantity $\pi_v := \pi(v, S_v; T, K, \sigma) \in [0,1]$, where the dependence on (T,K,σ) is suppressed when no confusion can arise. Thus π_v denotes the probability of choosing the stopping action at the

current time-state point (v, S_v). The cases $\pi_v = 0$ and $\pi_v = 1$ correspond to always continuing and always choosing the penalized stopping action, respectively. To encourage exploration during training, an entropy term is added so that the policy does not collapse to a purely stopping or purely continuation rule early in training.

We use the entropy term

$$H(\pi) = \pi \log \pi + (1 - \pi) \log(1 - \pi), \qquad \pi \in [0, 1],$$

with the continuous extension $H(0) = H(1) = 0$. The quantity $-H(\pi)$ is the Shannon entropy (Shannon, 1948) of the Bernoulli distribution, equal to 0 when the action is deterministic and is maximized at $\pi = \frac{1}{2}$. Therefore, the term $-\lambda H(\pi_v)$ rewards randomized decisions during training, where $\lambda > 0$ controls the strength of exploration.

For a fixed Bernoulli policy π, the premium-learning problem becomes a policy-evaluation problem of the form introduced in (3.1). Here the terminal payoff is zero, and the running reward is induced by the penalized stopping action and the entropy term. More precisely, at a future time $v \in [t, T]$, write

$$\pi_v := \pi(v, S_v; T, K, \sigma) \in [0, 1],$$

where the dependence on (T, K, σ) is suppressed when no confusion can arise. Thus π_v is the probability of choosing the penalized stopping action at the time-state point (v, S_v).

For a fixed policy π, the policy-dependent running reward is

$$\rho^{\pi}(v, S_v) := \Gamma \widehat{g}(v, S_v)\pi_v - \lambda H(\pi_v).$$

Here the first term is the penalty-induced contribution from the stopping action, while the second term is the entropy regularization contribution. This running term is not an additional payoff of the original option contract; it is the instantaneous term accumulated in the fixed-policy premium formulation.

The stopping probability also enters the discounting through the penalty-induced weighting. For a value function started at time t, define, for $v \in [t, T]$,

$$\frac{d}{dv} R^{\pi}_{t,v} = -\Gamma \pi_v R^{\pi}_{t,v}, \qquad R^{\pi}_{t,t} = 1. \tag{3.13}$$

The initial condition, $R^{\pi}_{t,t} = 1$, means that no penalty-driven weighting has accumulated at the starting time t. Solving this ordinary differential equation gives

$$R^{\pi}_{t,v} = \exp\left(-\Gamma \int_t^v \pi_\ell \, d\ell\right), \qquad v \in [t, T].$$

This factor is always positive and records the penalty-induced weighting accumulated from the starting time t to the future time v. Together with the financial discount factor, the total weighting becomes

$$\exp\left(-\int_t^v (r + \Gamma \pi_\ell) \, d\ell\right) = e^{-r(v-t)} R^{\pi}_{t,v}.$$

Thus future contributions to the premium value are discounted both by the risk-free rate and by the stopping intensity induced by the fixed Bernoulli policy.

Therefore, for a fixed Bernoulli policy π, the corresponding fixed-policy premium value is

$$J^{\pi}_{\widehat{E}}(t,s) := \mathbb{E}^{\mathbb{Q}}\left[\int_t^T e^{-r(v-t)} R^{\pi}_{t,v}\rho^{\pi}(v,S_v)\,dv \;\middle|\; S_t = s\right]. \tag{3.14}$$

The zero terminal condition of the premium is the reason that this representation contains only the accumulated running term and no separate terminal payoff.

The notation $J^{\pi}_{\widehat{E}}$, in equation 3.14, distinguishes the value generated by a fixed Bernoulli policy from the optimized premium value. At the level of the penalized and entropy-regularized problem, the target premium can be written as

$$U^{\lambda}_{A,\Gamma}(t,s) = \sup_{\pi} J^{\pi}_{\widehat{E}}(t,s),$$

where the supremum is taken over admissible Bernoulli stopping policies. In the implementation, U_{A,θ_A} is the neural approximation of this optimized premium, while $J^{\pi}_{\widehat{E}}$ is the fixed-policy value used during policy evaluation.

The fixed policy is then improved pointwise (Dai et al., 2025). For a value argument w, the local Hamiltonian is

$$\mathcal{H}_{\widehat{E}}(t,s,\pi;w) = -\Gamma\pi w + \Gamma\pi\widehat{g}(t,s) - \lambda H(\pi).$$

Here w is a placeholder for the current premium value at the time-state point (t,s). In the

exact fixed-policy problem, this value is $J^{\pi}_{\widehat{E}}(t,s)$. In the implementation, w is replaced by the current neural approximation $U_{A,\theta_A}(t,s)$. Maximizing the Hamiltonian over $\pi \in (0,1)$ gives the Bernoulli policy-improvement rule below(3.15). The one-dimensional maximization leading to this formula is given in Appendix B.2:

$$\pi^{*}_{\widehat{E}}(t,s;w) = \frac{1}{1+\exp(\lambda^{-1}\Gamma(w-\widehat{g}(t,s)))}. \tag{3.15}$$

Thus the stopping probability is large when the current premium estimate is below the baseline-adjusted exercise obstacle, and small when the premium estimate is above it.

We estimate the American option premium U_A by a neural network U_{A,θ_A}. In the same terminal-conditioned spirit as the European baseline, we write

$$U_{A,\theta_A}(t,s;T,K,\sigma) = \widehat{g}(t,s;T,K,\sigma) + (T-t)\mathcal{N}^{A}_{\theta_A}(t,s;T,K,\sigma), \tag{3.16}$$

where $\mathcal{N}^{A}_{\theta_A}$ is a scalar-output neural network.

The rule in (3.15) is a policy-improvement map for a given value argument w. In the implementation, the exact value argument is not available, so it is replaced by the current neural premium approximation. Thus, at a grid point (t_ℓ, S_{t_ℓ}), we set $\pi_\ell = \pi^{*}_{\widehat{E}}\left(t_\ell, S_{t_\ell}; U_{A,\theta_A}(t_\ell, S_{t_\ell})\right)$, that is,

$$\pi_\ell = \frac{1}{1+\exp(\lambda^{-1}\Gamma(U_{A,\theta_A}(t_\ell,S_{t_\ell}) - \widehat{g}(t_\ell,S_{t_\ell})))}.$$

Here the dependence on (T,K,σ) is suppressed for readability. Within one loss evaluation,

the resulting probabilities (π_ℓ) are treated as fixed pathwise quantities.

We now construct the offline martingale loss for the fixed policy. For the correct fixed-policy value $J^\pi_{\widehat{E}}$, define the reward-adjusted process

$$M^\pi_{t,v} := e^{-r(v-t)} R^\pi_{t,v} J^\pi_{\widehat{E}}(v, S_v) + \int_t^v e^{-r(\ell-t)} R^\pi_{t,\ell} \rho^\pi(\ell, S_\ell)\, d\ell, \qquad v \in [t, T]. \tag{3.17}$$

For the correct value function, $M^\pi_{t,v}$ is a martingale in the future time variable v. Replacing $J^\pi_{\widehat{E}}$ by the neural approximation U_{A,θ_A} gives

$$M^{\pi,\theta_A}_{t,v} := e^{-r(v-t)} R^\pi_{t,v} U_{A,\theta_A}(v, S_v) + \int_t^v e^{-r(\ell-t)} R^\pi_{t,\ell} \rho^\pi(\ell, S_\ell)\, d\ell. \tag{3.18}$$

This process is the martingale candidate used to measure the fixed-policy evaluation error.

Since the premium has zero terminal value, $J^\pi_{\widehat{E}}(T, S_T) = 0$, the terminal martingale value is the accumulated discounted running reward,

$$M^\pi_{t,T} = \int_t^T e^{-r(\ell-t)} R^\pi_{t,\ell} \rho^\pi(\ell, S_\ell)\, d\ell.$$

The offline American martingale loss is therefore

$$\mathcal{L}^{\text{off}}_{A,\pi}(\theta_A) := \frac{1}{2}\mathbb{E}^{\mathbb{Q}} \left[\int_t^T \left| M^\pi_{t,T} - M^{\pi,\theta_A}_{t,v} \right|^2 dv \;\middle|\; S_t = s \right]. \tag{3.19}$$

This compares the approximate reward-adjusted process at intermediate times with the terminal accumulated running reward under the fixed policy.

On the exercise grid $t = t_0 < t_1 < \cdots < t_L = T$, the penalty-induced weighting is updated by

$$R^{\pi}_{t,t_{j+1}} = R^{\pi}_{t,t_j}\left(1 - \Gamma\pi_{t_j}\Delta t\right), \qquad R^{\pi}_{t,t} = 1.$$

The discrete offline residual at grid time t_ℓ is

$$G^{\pi}_{\ell}(\theta_A) := \sum_{j=\ell}^{L-1} e^{-r(t_j-t)} R^{\pi}_{t,t_j}\rho^{\pi}(t_j, S_{t_j})\,\Delta t - e^{-r(t_\ell - t)} R^{\pi}_{t,t_\ell} U_{A,\theta_A}(t_\ell, S_{t_\ell}), \tag{3.20}$$

which is the time-discretized version of $M^{\pi}_{t,T} - M^{\pi,\theta_A}_{t,t_\ell}$. The first term is the accumulated discounted running reward from t_ℓ to maturity under the fixed policy, and the second term is the discounted premium predicted by the network at t_ℓ.

The implemented offline American loss is therefore

$$\widehat{\mathcal{L}}^{\text{off}}_{A,\pi}(\theta_A) := \frac{1}{2}\mathbb{E}^{\mathbb{Q}}\left[\sum_{\ell=0}^{L-1}\left(G^{\pi}_{\ell}(\theta_A)\right)^2 \Delta t \;\middle|\; S_t = s\right]. \tag{3.21}$$

In practice, the expectation is replaced by the empirical average over simulated risk-neutral paths. Each offline American update is therefore a fixed-policy evaluation step. After the optimizer step, the stopping probabilities are updated again through the logistic policy-improvement rule (3.15). The derivation of the reward-adjusted martingale property, the continuous-time loss, and the discrete residual $G^{\pi}_{\ell}(\theta_A)$ is given in Appendix B.3.

3.2 Online Learning

This section develops the online learning procedure. Unlike the offline method, which constructs its loss after a simulated path has been generated over the relevant time interval, the online method updates the parameters locally along the path. Each update uses information from two consecutive grid points and therefore has the form of a one-step TD method.

In classical RL, TD(0) (Sutton, 1988; Sutton and Barto, 2018) updates a value approximation by comparing the current value estimate with a one-step target formed from the next value estimate and the reward observed over the step. In the European option problem, there is no running reward before maturity, so the one-step target is the discounted next value. For the approximation $V_{E,\theta}$, the one-step residual is

$$\delta^{\theta}_{E,n} := e^{-r\Delta t_n} V_{E,\theta}(t_{n+1}, S_{t_{n+1}}) - V_{E,\theta}(t_n, S_{t_n}). \tag{3.22}$$

This residual measures the one-step violation of the discounted martingale condition.

The online method used here is based on martingale orthogonality (Jia and Zhou, 2022) rather than on minimizing the squared residual $(\delta^{\theta}_{E,n})^2$. As shown by Jia and Zhou (2022), the mean-square TD objective is not an appropriate value-error objective in continuous-time policy evaluation. Even for the true value function, the one-step martingale residual has zero conditional mean but generally nonzero conditional variance. Thus the squared residual is driven by martingale fluctuations, or quadratic variation in the continuous-time

limit, rather than by the mean-square error of the value function. A brief explanation of this point in the European option notation is given in Appendix B.4.

Since the true discounted European value process is a martingale, it satisfies

$$\mathbb{E}^{\mathbb{Q}}\left[e^{-r\Delta t_n}V_E(t_{n+1}, S_{t_{n+1}}) - V_E(t_n, S_{t_n}) \;\middle|\; \mathcal{F}_{t_n}\right] = 0.$$

The goal of the online method is not to force $\delta^{\theta}_{E,n}$ to be zero pathwise, but to enforce the martingale condition through suitable moment conditions.

For the approximation $V_{E,\theta}$, we enforce the martingale condition through a gradient-based moment condition. Following the martingale-orthogonality approach of Jia and Zhou (2022), we multiply the one-step residual by the current parameter sensitivity, the gradient of the approximated European value with respect to the trainable parameters. This gives

$$\mathbb{E}^{\mathbb{Q}}\left[\nabla_\theta V_{E,\theta}(t_n, S_{t_n})\, \delta^{\theta}_{E,n}\right] = 0. \tag{3.23}$$

Here $\nabla_\theta V_{E,\theta}(t_n, S_{t_n})$ is $\mathcal{F}_{t_n}$-measurable and connects the martingale residual directly to the parameters being updated.

The moment condition (3.23) suggests a stochastic update: on each sampled path and time step, we move the parameters in the direction $\nabla_\theta V_{E,\theta}(t_n, S_{t_n})\delta^{\theta}_{E,n}$. This gives the CTD(0)-type rule

$$\theta \leftarrow \theta + \alpha_n \nabla_\theta V_{E,\theta}(t_n, S_{t_n})\, \delta^{\theta}_{E,n}, \tag{3.24}$$

where $(\alpha_n)_{n=0}^{N-1}$ is a sequence of positive learning rates. Following the terminology of Jia

and Zhou (2022), this is called CTD(0) because it is a continuous-time analogue of TD(0).

In implementation, the semi-gradient update is obtained by detaching the residual from the computational graph. Equivalently, one may minimize the local surrogate objective

$$\ell^{\mathrm{on}}_{E,n}(\theta) = -\operatorname{sg}\big(\delta^{\theta}_{E,n}\big)\, V_{E,\theta}(t_n, S_{t_n}), \tag{3.25}$$

where $\operatorname{sg}(\cdot)$ denotes the stop-gradient operator. One gradient-descent step on (3.25) gives the CTD(0) semi-gradient update (3.24). The reason this is called a semi-gradient update is explained in Appendix B.5.

After the online European baseline has been pre-trained, the resulting approximation is again denoted by $\widehat{V}_E$. It is then held fixed and used as the baseline for the online American premium-learning problem.

We now describe the corresponding online update for the American premium. The same premium approximation U_{A,θ_A} from (3.16) is used. As in the offline American method, the Bernoulli stopping probabilities are first computed from the current premium approximation through the policy-improvement rule (3.15). During one update, these probabilities are treated as fixed, so the problem is again a fixed-policy evaluation problem.

For the correct fixed-policy premium value $J^{\pi}_{\widehat{E}}$, the reward-adjusted process $M^{\pi}_{t,v}$ in (3.17) is a martingale in the future time variable v. Therefore, on the exercise grid time,

$$\mathbb{E}^{\mathbb{Q}}\left[M^{\pi}_{t,t_{\ell+1}} - M^{\pi}_{t,t_\ell} \;\middle|\; \mathcal{F}_{t_\ell}\right] = 0.$$

The online American update enforces this one-step martingale condition after replacing $J^{\pi}_{\widehat{E}}$ by the neural premium approximation U_{A,θ_A}. For compactness, write

$$U_\ell := U_{A,\theta_A}(t_\ell, S_{t_\ell}), \qquad U_{\ell+1} := U_{A,\theta_A}(t_{\ell+1}, S_{t_{\ell+1}}), \qquad \rho^{\pi}_\ell := \rho^{\pi}(t_\ell, S_{t_\ell}).$$

Using the approximate reward-adjusted process $M^{\pi,\theta_A}_{t,v}$ from (3.18), removing the common positive factor $e^{-r(t_\ell - t)}R^{\pi}_{t,t_\ell}$, and using a first-order time discretization gives the online American residual

$$\delta^{\theta_A}_\ell := \frac{R^{\pi}_{t,t_{\ell+1}}}{R^{\pi}_{t,t_\ell}} U_{\ell+1} - U_\ell + (\rho^{\pi}_\ell - rU_\ell)\Delta t. \tag{3.26}$$

The derivation of the online American residual from the one-step increment of the reward-adjusted martingale is given in Appendix B.6.

Here the ratio

$$\frac{R^{\pi}_{t,t_{\ell+1}}}{R^{\pi}_{t,t_\ell}} = 1 - \Gamma\pi_{t_\ell}\Delta t.$$

is the penalty-induced weighting accumulated over the single time step $[t_\ell, t_{\ell+1}]$. The term $\rho^{\pi}_\ell \Delta t$ is the accumulated running term over this step, while $-rU_\ell\Delta t$ is the first-order contribution of financial discounting.

Following the same martingale-orthogonality principle as in the European online update, we test this residual against the current parameter sensitivity of the premium approximation. This gives the moment condition

$$\mathbb{E}^{\mathbb{Q}}\left[\nabla_{\theta_A} U_{A,\theta_A}(t_\ell, S_{t_\ell})\, \delta^{\theta_A}_\ell \;\middle|\; S_t = s\right] = 0. \tag{3.27}$$

The corresponding CTD(0)-type semi-gradient update is

$$\theta_A \leftarrow \theta_A + \alpha_\ell \nabla_{\theta_A} U_{A,\theta_A}(t_\ell, S_{t_\ell})\, \delta_\ell^{\theta_A}, \tag{3.28}$$

where $\alpha_\ell > 0$ is the learning rate. As in the European case, the residual is treated as fixed during differentiation. Equivalently, the update can be implemented by minimizing the local surrogate objective

$$\ell_\ell^{\mathrm{on}}(\theta_A) = -\operatorname{sg}\Big(\delta_\ell^{\theta_A}\Big)\, U_{A,\theta_A}(t_\ell, S_{t_\ell}). \tag{3.29}$$

One gradient-descent step on this surrogate gives the update (3.28). The derivation of the one-step American residual in (3.26) can be obtained from the reward-adjusted martingale increments, in the same way as the offline residual in Appendix B.3.

After learning, the randomized Bernoulli policy is not used directly for pricing. Randomization is used only during training. The final pricing rule is deterministic. From (3.15),

$$\pi^*_{\widehat{E}}(t, s; w) \geq \frac{1}{2} \quad \Longleftrightarrow \quad w \leq \widehat{g}(t, s).$$

Replacing w by the learned premium approximation gives the deterministic execution rule

$$u_{\theta_A}(t, s) := \mathbf{1}_{\{U_{A,\theta_A}(t,s) \leq \widehat{g}(t,s)\}}. \tag{3.30}$$

Thus exercise is recommended when the learned premium is no larger than the baseline-

adjusted exercise obstacle.

On the exercise grid, the induced stopping time is

$$\tau_{\theta_A} := \inf\{t_\ell : U_{A,\theta_A}(t_\ell, S_{t_\ell}) \le \widehat{g}(t_\ell, S_{t_\ell})\} \wedge T. \tag{3.31}$$

The stopping-based American price reported in the numerical results is

$$P_{\text{stop}} := \mathbb{E}^{\mathbb{Q}}\left[e^{-r(\tau_{\theta_A}-t)} g_K(S_{\tau_{\theta_A}}) \;\middle|\; S_t = s\right]. \tag{3.32}$$

Because exercise is checked only on the grid, this execution rule is a Bermudan approximation of the continuous-time American stopping problem.

3.3 Specific Moderate-Jump Merton Extension

The offline and online learning procedures above were formulated under Black–Scholes geometric Brownian motion dynamics. In the numerical study, we retain their high-level two-stage structure under the Merton jump–diffusion model introduced in Section 2.5. For each learning mode, a European baseline is first trained from simulated risk-neutral Merton paths and then held fixed while the American premium is learned relative to it. The GBM simulator and transition law are replaced by their Merton counterparts. Whenever the infinitesimal generator enters the analytical formulation, the Black–Scholes generator $\mathcal{L}$ is replaced by the Merton generator $\mathcal{L}_M$ in (2.13).

Let $\widehat{V}_E^M(t,s;T,K,\sigma,\eta_J,\mu_J,\delta_J)$ denote the learned European value under the Merton model. The corresponding baseline-adjusted exercise obstacle, denoted by $\widehat{g}^M$, is defined as the Merton counterpart of (3.9), with $\widehat{V}_E$ replaced by $\widehat{V}_E^M$. Thus, the learned-baseline decomposition and exercise comparison retain the same algebraic form as under GBM. The American network learns a correction relative to the learned Merton baseline. Because the baseline and American network are approximate, this correction need not coincide exactly with the economic early-exercise premium $V_A^M - V_E^M$.

The penalty formulation, entropy regularization, Bernoulli policy-improvement formula in (3.15), and deterministic execution rule in (3.30) follow Dai et al. (2025). The distinction between the whole-trajectory martingale-loss objective and the one-step CTD(0)-type learning rule derived from martingale orthogonality is also retained (Jia and Zhou, 2022; Dai et al., 2025). The final Merton implementation uses the same forward-Euler recurrence as the Black–Scholes implementation:

$$R_{n+1} = R_n\left(1 - \Gamma\pi_n\Delta t\right), \qquad R_0 = 1. \tag{3.33}$$

This numerical discretization does not change the underlying penalized stopping formulation.

Both final Merton American learning procedures account for the martingale defect of the frozen European network. Suppressing the contract and Merton-parameter arguments, if the frozen European value were exact, the conditional mean, given the information at

t_n, of

$$e^{-r\Delta t}\widehat{V}_E^M(t_{n+1}, S_{t_{n+1}}) - \widehat{V}_E^M(t_n, S_{t_n})$$

would be zero. For the learned baseline, this conditional mean need not vanish. Using the survival update in (3.33), the terms added to the two American residuals are

$$\begin{aligned} &\text{offline:} \quad \sum_{j=n}^{L-1} e^{-rt_j} R_{j+1} \left[e^{-r\Delta t}\widehat{V}_E^M(t_{j+1}, S_{t_{j+1}}) - \widehat{V}_E^M(t_j, S_{t_j}) \right], \\ &\text{online:} \quad (1 - \Gamma\pi_n\Delta t) \left[e^{-r\Delta t}\widehat{V}_E^M(t_{n+1}, S_{t_{n+1}}) - \widehat{V}_E^M(t_n, S_{t_n}) \right]. \end{aligned} \tag{3.34}$$

The final Merton online implementation uses the exact one-step financial discount in the premium term. Let

$$a_n := \frac{R_{n+1}}{R_n} = 1 - \Gamma\pi_n\Delta t, \qquad d := e^{-r\Delta t}, \qquad U_n^M := U_{A,\theta_A}^M(t_n, S_{t_n}).$$

The complete Merton online residual is

$$\delta_n^M = d\, a_n U_{n+1}^M - U_n^M + \rho_n^\pi \Delta t + a_n \left[d\, \widehat{V}_E^M(t_{n+1}, S_{t_{n+1}}) - \widehat{V}_E^M(t_n, S_{t_n}) \right].$$

The offline procedure adds the complete future tail to its full-return residual, without an additional factor of Δt, whereas the procedure labelled online adds only the corresponding one-step term to its CTD(0)-type residual. These additions make the sampled residuals algebraically correspond to the reconstructed value $\widehat{V}_E^M + U_{A,\theta_A}^M$. They do not make the frozen European baseline exact or remove sampling, approximation, or optimization errors.

The numerical organization of the updates is also adapted. For each learning mode, the European network is trained for 300 iterations. Each iteration samples eight distinct Merton specifications and pairs them with contracts using a rotating queue that keeps the nine contracts equally represented across successive iterations. Eight decision dates are then sampled uniformly for each specification–contract pair. The resulting gradients are averaged before one European optimizer update. An American outer sweep visits all nine contracts once, in random order. For each contract, contributions from four sampled Merton specifications and eight dates selected by a rotating balanced schedule are combined before one optimizer update. In the procedure labelled online, each selected date uses the consecutive pair (t_n, t_{n+1}), but these one-step contributions are batched before the update. Both American procedures therefore make nine optimizer updates per outer sweep, or 2700 updates over 300 outer sweeps. The Merton labels refer to the whole-trajectory full-return procedure and the batched one-step CTD(0)-type procedure; they do not describe different optimizer-update frequencies. The two modes nevertheless retain separately trained networks and their respective learning settings.

The penalty factor remains $\Gamma = 10$. By the obstacle definition in (3.9), the Merton exercise gap satisfies

$$U^M_{A,\theta_A} - \widehat{g}^M = (\widehat{V}^M_E + U^M_{A,\theta_A}) - g_K. \tag{3.35}$$

Hence, it is the difference between the reconstructed American value and the immediate-exercise payoff. By the deterministic rule in (3.30), a positive gap gives continuation, whereas a nonpositive gap gives exercise at a pre-maturity decision date.

For the final Merton implementation, the parameterization in (3.16) is replaced, for $t < T$, by

$$U^M_{A,\theta_A}(t,s) - \widehat{g}^M(t,s) = c_{\text{gap}} z_{\theta_A}(t,s). \tag{3.36}$$

Here, z_{θ_A} is the raw scalar network output, and the zero terminal condition is imposed separately. The positive scale is chosen as

$$c_{\text{gap}} = \frac{\lambda_0}{\Gamma} = \frac{0.025}{10} = 0.0025. \tag{3.37}$$

Substitution of (3.36) into the Bernoulli rule (3.15) gives the coefficient $\Gamma c_{\text{gap}}/\lambda$ on z_{θ_A} in the logistic exponent. This coefficient equals one at the initial temperature $\lambda_0 = 0.025$. The temperature is then reduced over successive thirds of American training according to

$$\lambda \in \{0.025, 0.005, 0.001\}. \tag{3.38}$$

This schedule gradually sharpens the randomized exercise comparison. Since $c_{\text{gap}} > 0$, the parameterization preserves the sign of the gap for a fixed network output, and the deterministic threshold is $z_{\theta_A} \leq 0$.

The training paths use broader initial-state coverage than in the original GBM implementation. The base proposal used for the European batches and the American candidate banks places 20% of the states at S_0 and draws the remaining 80% by stratified log-uniform sampling over $[0.495S_0, 1.32S_0]$. In the American stage, candidate starts are further reallocated among time-zero policy-threshold, European-gap, downside, and ordinary strata

to increase coverage of exercise-relevant states. Importance weights correct for this reallocation in the learning objective. The first ten American outer sweeps use the initial baseline-based policy. The critic is then copied to a frozen policy snapshot, which is refreshed after every ten complete outer sweeps.

In addition to the state and contract features defined above, the Merton networks receive $\sigma, \eta_J, \mu_J, \delta_J$ and the derived jump features $\eta_J\kappa$ and $\eta_J(\mu_J^2 + \delta_J^2)$, where κ is the Merton jump compensator introduced earlier. Both networks use two hidden layers of widths 128 and 64, respectively, with rectified linear unit (ReLU) activations and without batch normalization. The jump parameters are not estimated or calibrated during training. Instead, they are sampled from a curated pool of 31 moderate-jump specifications and supplied as network inputs, so the networks learn dependence on the retained parameter family.

The European benchmark is computed from the Merton series formula in (2.16). The American benchmark is a state-space Bermudan dynamic-programming approximation on the same exercise grid, using the Merton transition law. These benchmark values, continuation values, and exercise boundaries are used only for evaluation and do not enter training.

The reported numerical table evaluates the selected retained specification $(\sigma, \eta_J, \mu_J, \delta_J) = (0.25, 0.50, -0.10, 0.20)$, with $\kappa = -0.076884$ on independent paths. The experiment is therefore a restricted robustness test of the learned-baseline framework after model-specific numerical adaptations, rather than a stress test over extreme or out-of-pool jump regimes.

The transferability of the high-level construction should not be interpreted as a universal performance guarantee. Other risk-neutral asset-price models may require changes to the simulator, Markov state, network inputs and parameterization, training design, and benchmark or calibration procedure. The present thesis therefore provides numerical evidence for the tested Black–Scholes contracts and one selected moderate-jump Merton specification after training over the retained pool; it does not establish uniform accuracy across the complete pool or other asset-price model classes.

4

Numerical Results

This chapter evaluates the martingale-based learning framework developed in Chapter 3. The aim is to test whether the pre-trained European baseline and the American premium-learning component produce accurate prices and reliable stopping rules when trained from simulated risk-neutral paths.

The numerical study is organized around two asset-price dynamics. The Black–Scholes

setting provides the controlled diffusion case, where stock paths are continuous. The moderate-jump Merton setting introduces finite-activity jumps and tests whether the same learned-baseline premium structure remains stable when the state dynamics are changed. For European puts, benchmark prices are computed from the corresponding analytic formulas: the Black–Scholes formula in the diffusion case and the Merton series formula in the jump–diffusion case. For American puts, benchmark prices are computed numerically on the exercise grid using a state-space dynamic-programming Bermudan approximation.

Throughout the chapter, the main American quantity of interest is the stopping-based price induced by the learned deterministic execution rule. This price is obtained by executing the learned stopping rule on simulated paths after exploration has ended. Therefore, the American results should be interpreted not only as value-approximation tests, but also as tests of whether the learned premium surface produces accurate exercise-versus-continuation decisions.

4.1 Black–Scholes Model

The first set of experiments is conducted under the Black–Scholes risk-neutral dynamics. The contract universe contains nine put contracts formed from three maturities and three moneyness levels. With $S_0 = 1$ and $\Delta t = 0.01$, the maturities are $T \in \{0.25, 0.50, 1.00\}$, and the strikes are $K \in \{1.1S_0, S_0, 0.9S_0\}$, corresponding respectively to in-the-money (ITM), at-the-money (ATM), and out-of-the-money (OTM) puts at the initial time.

The experiment uses shared contract-conditioned networks rather than one separate network per contract. For each learning mode, one European network is pre-trained across all maturities, strikes, and volatility values. After this network is frozen, one American premium network is trained across the same contract family. Volatility is treated as a model input: during training, $\sigma \in \{0.10, 0.15, \ldots, 0.60\}$, while the reported benchmark comparisons are evaluated at $\sigma = 0.30$.

The Black–Scholes neural-network architecture is kept simple. Both the European option price network and the American option premium network are fully connected feed-forward networks with two hidden layers containing 128 and 64 units, respectively, followed by one scalar output. ReLU activations are used in the hidden layers, and batch normalization is applied after each hidden layer. The networks are contract-conditioned. Their inputs include the current time and state together with the relevant contract and model variables. The Merton networks retain the same hidden-layer widths and ReLU activations. They do not use batch normalization and additionally receive the jump–diffusion parameters and derived jump features.

For each sampled volatility, stock paths are generated from the exact log-Euler (Glasserman, 2004) form

$$S_{t_{n+1}} = S_{t_n} \exp\left(\left(r - q - \frac{1}{2}\sigma^2\right)\Delta t + \sigma\sqrt{\Delta t}\, Z_{n+1}\right), \qquad Z_{n+1} \sim N(0,1).$$

Each training batch is simulated up to the longest maturity $T = 1$, and shorter-maturity contracts use the corresponding path prefixes.

This design makes the experiment more demanding than a single-contract fit. The reported prices are produced by shared learned surfaces, so the results test whether the martingale-based objectives learn a reusable pricing and stopping representation across maturity, moneyness, volatility, time, and state.

Table 4.1 shows that the American stopping-based prices are more stable than the direct European value-surface estimates. In Panel A, the European price $\widehat{V}_0^E$ is obtained by evaluating the pre-trained European value surface at the initial state, $\widehat{V}_0^E = \widehat{V}_E(0, S_0; T, K, \sigma)$.

In Panel B, however, $P_{\text{stop},0}$ is not a direct network readout at $(0, S_0)$. It is the discounted payoff obtained by executing the deterministic stopping rule induced by the learned American premium. Thus the European and American errors are not generated by the same mechanism: the European error measures pointwise value-surface accuracy, while the American error measures the quality of the induced stopping rule.

The European option pricing errors show a general moneyness pattern. The largest relative errors occur mostly for OTM contracts, for which the benchmark European option values are smaller and a moderate absolute pricing deviation can therefore produce a larger percentage error. For example, the offline European option value learner has OTM relative errors of 8.40%, 8.90%, and 5.49% for $T = 0.25, 0.50, 1.00$, respectively. The online European option value learner has corresponding OTM relative errors of 3.41%, 7.81%, and 8.01%. However, for $T = 0.25$, the ATM relative error is larger than the OTM relative error under both learning procedures. These relative errors are therefore best interpreted together with the scale of the corresponding benchmark European option value.

Table 4.1: Black–Scholes pricing and Greek errors for the offline and online learning procedures.

Panel A: European put options

		Initial-time pricing				Grid-based L^2 errors				
T	Mon.	$V_0^{E,*}$	$\widehat{V}_0^E$	$V_0^{E,*} - \widehat{V}_0^E$	Rel. (%)	Price	Delta	Gamma	Vega	Theta
Offline learning										
0.25	ATM	0.058471	0.053468	0.0050024	8.56	0.00000936	0.0027039	1.767103	0.0003761	0.0003616
0.25	ITM	0.122919	0.123919	−0.0010003	0.81	0.00001666	0.0020048	1.596394	0.0004043	0.0005111
0.25	OTM	0.019627	0.021277	−0.0016496	8.40	0.00000593	0.0057033	1.999204	0.0002850	0.0003257
0.50	ATM	0.081789	0.077915	0.0038742	4.74	0.00002025	0.0043693	1.277377	0.0007943	0.0009779
0.50	ITM	0.143580	0.140704	0.0028758	2.00	0.00003242	0.0045744	1.137358	0.0005915	0.0010435
0.50	OTM	0.038356	0.041768	−0.0034130	8.90	0.00001476	0.0058756	1.441608	0.0009491	0.0008565
1.00	ATM	0.113733	0.113342	0.0003908	0.34	0.00010398	0.0062311	0.899279	0.0016174	0.0006711
1.00	ITM	0.174033	0.170785	0.0032477	1.87	0.00024147	0.0077013	0.799755	0.0023720	0.0032267
1.00	OTM	0.066423	0.070071	−0.0036475	5.49	0.00002098	0.0068113	0.992669	0.0023791	0.0017503
Online learning										
0.25	ATM	0.058471	0.061919	−0.0034482	5.90	0.00000642	0.0019979	1.607957	0.0001685	0.0001973
0.25	ITM	0.122919	0.121524	0.0013946	1.13	0.00001367	0.0021286	1.467419	0.0002454	0.0003652
0.25	OTM	0.019627	0.018959	0.0006685	3.41	0.00000548	0.0030031	1.835655	0.0001918	0.0002179
0.50	ATM	0.081789	0.082821	−0.0010313	1.26	0.00002038	0.0021099	1.170220	0.0004488	0.0003660
0.50	ITM	0.143580	0.139584	0.0039957	2.78	0.00004344	0.0026177	1.049350	0.0008076	0.0005549
0.50	OTM	0.038356	0.035358	0.0029973	7.81	0.00001766	0.0037488	1.321667	0.0003340	0.0002679
1.00	ATM	0.113733	0.112058	0.0016749	1.47	0.00005668	0.0053110	0.843187	0.0023606	0.0003122
1.00	ITM	0.174033	0.167316	0.0067164	3.86	0.00013679	0.0053525	0.747263	0.0046881	0.0004794
1.00	OTM	0.066423	0.061100	0.0053232	8.01	0.00005029	0.0050797	0.991284	0.0035429	0.0002570

Panel B: American put options

		Initial-time pricing				Execution-based L^2 errors				
T	Mon.	$V_0^{A,\mathrm{ref}}$	$P_{\mathrm{stop},0}$	$V_0^{A,\mathrm{ref}} - P_{\mathrm{stop},0}$	Rel. (%)	Price	Delta	Gamma	Vega	Theta
Offline learning										
0.25	ATM	0.058704	0.059338	−0.0006341	1.08	0.00057774	0.0133457	517745.553	0.0001245	0.0002634
0.25	ITM	0.123407	0.123646	−0.0002393	0.19	0.00000132	0.0000480	0.029	0.0000627	0.0006483
0.25	OTM	0.019758	0.019508	0.0002504	1.27	0.00102722	0.0062767	80045.315	0.0008494	0.0018254
0.50	ATM	0.082371	0.080758	0.0016140	1.96	0.00062970	0.0103712	576192.987	0.0004699	0.0005444
0.50	ITM	0.144534	0.142189	0.0023452	1.62	0.00001142	0.0000912	0.032	0.0000512	0.0006763
0.50	OTM	0.038745	0.037814	0.0009311	2.40	0.00017255	0.0050310	123714.839	0.0002612	0.0028379
1.00	ATM	0.115217	0.114622	0.0005951	0.52	0.00076846	0.0080352	699171.251	0.0039846	0.0004412
1.00	ITM	0.176090	0.173358	0.0027316	1.55	0.00000924	0.0003026	0.048	0.0003656	0.0004488
1.00	OTM	0.067548	0.066540	0.0010076	1.49	0.00026641	0.0042308	201767.369	0.0001071	0.0014665
Online learning										
0.25	ATM	0.058704	0.058708	−0.0000034	0.006	0.00056801	0.0131371	517852.964	0.0001097	0.0003656
0.25	ITM	0.123407	0.123209	0.0001973	0.16	0.00000323	0.0000698	0.040	0.0000503	0.0041415
0.25	OTM	0.019758	0.019171	0.0005873	2.97	0.00010143	0.0062416	80014.604	0.0008671	0.0028165
0.50	ATM	0.082371	0.080325	0.0020462	2.48	0.00062509	0.0107392	576202.533	0.0004410	0.0014271
0.50	ITM	0.144534	0.143753	0.0007803	0.54	0.00001394	0.0001625	0.043	0.0001225	0.0008549
0.50	OTM	0.038745	0.038710	0.0000354	0.09	0.00016197	0.0053149	123670.912	0.0003190	0.0003767
1.00	ATM	0.115217	0.113549	0.0016682	1.45	0.00078114	0.0077365	699156.889	0.0042123	0.0064407
1.00	ITM	0.176090	0.174875	0.0012146	0.69	0.00000472	0.0021179	0.047	0.0004431	0.0026916
1.00	OTM	0.067548	0.066978	0.0005705	0.84	0.00026265	0.0043380	201820.098	0.0002413	0.0018073

Notes. Panel A reports European put prices and grid-based price and Greek errors. Panel B reports American put prices and the corresponding execution-based grid errors. Mon. denotes moneyness: ITM, ATM, and OTM. In Panel A, Rel. is $|V_0^{E,*} - \widehat{V}_0^E|/V_0^{E,*}$; in Panel B, Rel. is $|V_0^{A,\mathrm{ref}} - P_{\mathrm{stop},0}|/V_0^{A,\mathrm{ref}}$. Both are reported as percentages, and the adjacent difference columns report signed pricing errors. The parameters are $S_0 = 1$, $r = 0.01$, $q = 0$, $\sigma = 0.30$, $\Delta t = 0.01$, and $T \in \{0.25, 0.50, 1.00\}$. The strikes are $K = 1.1$, $K = 1.0$, and $K = 0.9$ for ITM, ATM, and OTM puts, respectively. Each stage uses 1000 training iterations with a training path batch size of 2048. The American initial stopping-price estimate uses 10000 evaluation paths, whereas the execution-based grid and Greek estimates use 2000 common-random-number paths.

The sign pattern shows that the European learner does not have a single global bias. With the convention $V_0^{E,*} - \widehat{V}_0^E > 0$, a positive error means that the learned European price is below the benchmark. The offline European learner overestimates all three OTM contracts but underestimates the ATM contracts. The online learner has a different pattern: most entries are underestimated, especially at $T = 1.00$, while the short- and medium-maturity ATM contracts are overestimated. Hence the European error depends on both maturity and moneyness.

The American stopping-based prices are substantially more accurate on average. The mean European relative error is 4.57% for the offline method and 3.96% for the online method, respectively. The corresponding American mean errors decrease to 1.34% and 1.03%, respectively. This improvement is not driven by a single contract. For the online $T = 0.50$ OTM put, the European error is 7.81%, while the corresponding American stopping-based error is 0.09%. For the online $T = 1.00$ OTM put, the European error is 8.01%, while the American error is 0.84%. The online $T = 0.25$ ATM contract is even more striking: the European error is 5.90%, but the American stopping-based error is 0.006%.

This behaviour is consistent with the learned-baseline premium construction. The pre-trained European option price network enters the American option premium-learning stage through the baseline-adjusted exercise obstacle, but the final American option price is computed by executing the learned stopping rule. Therefore, European value-surface error does not mechanically pass through to the American stopping-based price. What matters most for $P_{\text{stop},0}$ is whether the premium comparison produces accurate exercise-versus-continuation decisions along the simulated paths.

The signs of the American errors support this interpretation. With the Panel B convention, $V_0^{A,\mathrm{ref}} - P_{\mathrm{stop},0} > 0$ means that the learned execution rule prices below the numerical optimal-stopping benchmark. Since a fixed learned stopping rule is generally expected to perform no better than the benchmark stopping rule, positive signed errors are natural, apart from Monte Carlo noise and benchmark discretization effects. The online American learner underprices the benchmark in eight of the nine contracts, with the only negative error occurring for the $T = 0.25$ ATM contract, where the relative error is only 0.006%.

The sign of the European error does not determine the sign of the American error. For example, the offline European OTM contracts are overestimated for all three maturities, but the corresponding American stopping-based prices are underestimated. Similarly, the offline $T = 0.25$ ATM European price is underestimated by 8.56%, whereas the American stopping-based price is overestimated by only 1.08%. These sign reversals show that the American premium-learning component is not simply transmitting the European pricing bias.

The remaining larger American errors are moderate and interpretable. The online $T = 0.25$ OTM case has relative error 2.97%, and the online $T = 0.50$ ATM case has relative error 2.48%. These are cases where the option value is small or where the exercise decision is close to the continuation boundary. In such regions, small local errors in the learned premium comparison can move a noticeable fraction of paths across the stopping boundary.

The Greek errors should be interpreted separately from the stopping-price errors. In

particular, the large American gamma errors are expected, because gamma is a second derivative and is highly sensitive to the payoff kink and to nonsmoothness near the early-exercise boundary. The stopping-based prices remain accurate despite these local derivative errors, which suggests that the learned surfaces do not need to provide uniformly accurate higher-order sensitivities in order to generate useful stopping decisions.

Overall, Table 4.1 supports the effectiveness of the two-stage learned-baseline construction. The European learner provides a reusable contract-conditioned baseline, and the American premium learner converts that baseline into an execution rule. The resulting American prices are not dominated by the larger European value-surface errors; rather, the stopping-based evaluation indicates that the learned premium surface identifies exercise decisions accurately across the tested maturities and moneyness levels.

4.2 Merton Jump–Diffusion Dynamics

The second set of experiments applies the same two-stage learning pipeline to the Merton jump–diffusion dynamics introduced in Section 2.5. The purpose is not to redesign the learning method for jumps, but to test whether the pre-trained European baseline and the American premium-learning component remain stable when the stock path contains finite-activity discontinuities. The two learning modes keep their definitions from Chapter 3: the offline method uses the full future return, whereas the online method uses one-step CTD(0) residuals. These martingale-learning and randomized-stopping constructions follow the approaches of Jia and Zhou (2022) and Dai et al. (2025). The nine maturity–moneyness

contracts, penalty factor, and deterministic execution rule are also kept the same as in the Black–Scholes experiment. The path simulator, network conditioning, and benchmark calculations change, while the numerical training schedule is adapted to the more demanding Merton setting.

The Merton experiment is parameter-conditioned. The networks are not trained under one fixed jump specification only. Instead, each training iteration samples from a curated moderate-jump pool. Candidate parameters are taken from

$$\sigma \in \{0.15, 0.20, 0.25, 0.30\}, \qquad \eta_J \in \{0.50, 0.75\},$$

$$\mu_J \in \{-0.12, -0.10, -0.08\}, \qquad \delta_J \in \{0.15, 0.20\}.$$

Only parameter combinations satisfying

$$-0.12 \leq \kappa \leq -0.05, \qquad 0.24 \leq \sqrt{\sigma^2 + \eta_J(\mu_J^2 + \delta_J^2)} \leq 0.34$$

are retained, where κ, defined in (2.12), is the Merton jump compensator. The first condition keeps the jump component within a moderate downside range, while the second controls the combined diffusion–jump volatility scale. The retained pool contains 31 Merton specifications. At each training iteration of the European stage, 8 specifications are sampled from this pool and paired with contracts so that the nine contracts are used in turn. Each parameter–contract pair uses a separate training batch and 8 sampled dates. Their gradients are then averaged before one network update is applied.

The reported table is evaluated at the representative retained specification

$$(\sigma, \eta_J, \mu_J, \delta_J) = (0.25, 0.50, -0.10, 0.20), \qquad \kappa = -0.076884.$$

Thus the reported results should be read as evaluation on one representative moderate-jump case after training over a family of moderate jump–diffusion specifications.

The European component is pre-trained from simulated Merton paths and is not fitted to benchmark prices. The same offline and online distinction is retained as in the Black–Scholes experiment, with the simulated paths replaced by jump–diffusion paths and the network inputs augmented by $\sigma, \eta_J, \mu_J, \delta_J$ and derived jump features. The Merton networks use two ReLU hidden layers of widths 128 and 64. Unlike the Black–Scholes networks, batch normalization is not used in this implementation. For evaluation, the European reference values are computed from the Merton series formula in (2.16).

The American premium-learning component follows the same learned-baseline construction as in the Black–Scholes experiment. After the European Merton network is pre-trained, it is frozen and used to define the baseline-adjusted exercise obstacle

$$\widehat{g}^M(t, s; T, K, \sigma, \eta_J, \mu_J, \delta_J) = g_K(s) - \widehat{V}_E^M(t, s; T, K, \sigma, \eta_J, \mu_J, \delta_J).$$

This is the same form of exercise obstacle used in the Black–Scholes experiment.

One additional adjustment is needed because the frozen European network is only an approximation. Its discounted value need not satisfy the one-step martingale relation

exactly under the simulated Merton transition. The resulting residual is referred to here as the martingale defect of the frozen European option price baseline. The offline American loss includes the accumulated future defect, while the online American loss includes the corresponding one-step defect. This accounts for the part of the training residual arising from approximation error in the frozen baseline. Consequently, U^M_{A,θ_A} is a learned correction relative to that baseline and need not coincide exactly with the economic early-exercise premium $V^M_A - V^M_E$, which is defined relative to the Merton series value. The correction uses only the frozen network and simulated paths; neither the Merton series value nor the dynamic-programming benchmark is used during training.

Simply replacing the GBM paths by Merton paths while retaining the numerical training strategy used in the Black–Scholes experiment was not sufficient. A single Merton network must represent 31 specifications involving four model parameters, rather than variation in the diffusion volatility only. Moreover, at S_0 the benchmark early-exercise premium, $V_0^{A,\mathrm{ref}} - V_0^{E,*}$, is only about 0.5% to 1.8% of the total American value across the nine contracts. With training paths concentrated near S_0, this stopping signal was small relative to the full pricing problem. In preliminary implementations, the update schedule did not provide sufficient balanced coverage of the contracts and exercise-relevant states, and the learned rules continued most paths to maturity. These preliminary diagnostic results indicated that the issue was the coverage and organization of the optimization updates, rather than a change in the underlying stopping formulation.

The final Merton implementation broadens the range of initial stock values used during training. In the American stage, a larger set of candidate starting states is formed, and

the training sample gives greater coverage to downside starts and starts near the current exercise–continuation threshold. The associated loss weights account for this reallocation. An American outer sweep is defined as one complete pass through all nine contracts, visited in random order. For each contract, gradients from 4 sampled Merton specifications and 8 exercise dates are combined before one optimizer update is applied. Consequently, 300 outer sweeps give each contract 300 updates, or 2700 American optimizer updates in total for each learning mode. These changes balance the training information across contracts while preserving the offline full-return and online one-step distinction.

The difference between the learned American premium value and the exercise obstacle, $U^M_{A,\theta_A} - \widehat{g}^M$, determines the exercise decision. This quantity is referred to as the decision gap. For $t < T$, the final implementation writes

$$U^M_{A,\theta_A}(t,s) = \widehat{g}^M(t,s) + c_{\text{gap}} z_{\theta_A}(t,s), \qquad c_{\text{gap}} = 0.0025,$$

where z_{θ_A} is the raw network output and c_{gap} is a fixed positive scale factor. The terminal condition $U^M_{A,\theta_A}(T,s) = 0$ is imposed separately. Since $c_{\text{gap}} > 0$, this scaling does not change the sign of the exercise comparison.

The entropy temperature λ controls how gradual the randomized exercise comparison is during training: a larger value produces a smoother probability, whereas a smaller value makes the comparison sharper. Because the Merton decision gap is written on the scale $c_{\text{gap}} = 0.0025$, retaining the Black–Scholes value $\lambda = 1$ would make the exercise probabilities vary very little for raw network outputs near their initial scale and would

weaken the training signal. The penalty factor remains $\Gamma = 10$, while the temperature is reduced blockwise through

$$\lambda \in \{0.025, 0.005, 0.001\}$$

over successive thirds of training. This gives a smoother randomized comparison initially and a sharper exercise comparison later. The Bernoulli stopping probability, denoted by π, is evaluated directly in the loss rather than by drawing a Bernoulli action. At evaluation, the reported American price is generated by exercising at the first pre-maturity date for which $\pi \geq 0.5$, and otherwise at maturity.

The American benchmark is computed by the same state-space dynamic-programming Bermudan approximation used in the Black–Scholes study, with the Merton transition law replacing the GBM transition. The benchmark Greeks are obtained from the same numerical benchmark by finite-difference perturbations. For the reported table, the benchmark uses 601 log-stock grid points and 12-point Gauss–Hermite quadrature, a weighted integration rule for the Gaussian part of the one-step transition.

Therefore, the table below should be read as a robustness test of the same two-stage martingale-learning framework under finite-activity jump dynamics. The learning rules retain their Chapter 3 interpretation, while the state coverage, update organization, and temperature scale are adapted to the Merton jump–diffusion setting.

Table 4.2: Moderate-jump Merton pricing and Greek errors for the offline and online learning procedures.

Panel A: European put options

		Initial-time pricing				Grid-based L^2 errors				
T	Mon.	$V_0^{E,*}$	$\widehat{V}_0^E$	$V_0^{E,*} - \widehat{V}_0^E$	Rel. (%)	Price	Delta	Gamma	Vega	Theta
Offline learning										
0.25	ATM	0.0544152	0.0386438	0.01577130	28.98	0.00002092	0.0105684	1.999490	0.0015117	0.0029686
0.25	ITM	0.1188710	0.1208940	−0.00202259	1.70	0.00002650	0.0139308	1.817680	0.0018631	0.0025237
0.25	OTM	0.0184260	0.0185066	−0.00008060	0.44	0.00002075	0.0124472	2.236550	0.0013902	0.0089878
0.50	ATM	0.0773075	0.0596293	0.01767820	22.87	0.00006532	0.0074483	1.345250	0.0028786	0.0040259
0.50	ITM	0.1384380	0.1309120	0.00752635	5.44	0.00005819	0.0084866	1.191970	0.0044308	0.0024636
0.50	OTM	0.0361852	0.0270554	0.00912982	25.23	0.00011790	0.0068734	1.540870	0.0049614	0.0029976
1.00	ATM	0.1087070	0.0786832	0.03002410	27.62	0.00028651	0.0186949	0.930057	0.0144824	0.0089251
1.00	ITM	0.1680630	0.1451770	0.02288650	13.62	0.00038291	0.0142861	0.735865	0.0093602	0.0079416
1.00	OTM	0.0632089	0.0462408	0.01696810	26.84	0.00040393	0.0216744	1.026690	0.0060708	0.0176113
Online learning										
0.25	ATM	0.0544152	0.0425939	0.01182130	21.72	0.00006786	0.0113918	1.912340	0.0019584	0.0028922
0.25	ITM	0.1188710	0.1203710	−0.00149922	1.26	0.00007404	0.0145947	1.773850	0.0036957	0.0025957
0.25	OTM	0.0184260	0.0238199	−0.00539396	29.27	0.00010135	0.0107146	2.140430	0.0012012	0.0025540
0.50	ATM	0.0773075	0.0709576	0.00634988	8.21	0.00007853	0.0100925	1.197390	0.0036753	0.0078452
0.50	ITM	0.1384380	0.1446140	−0.00617631	4.46	0.00011769	0.0111876	1.074650	0.0027264	0.0107795
0.50	OTM	0.0361852	0.0393111	−0.00312590	8.64	0.00026955	0.0158852	1.312530	0.0046882	0.0022763
1.00	ATM	0.1087070	0.1288350	−0.02012810	18.52	0.00025493	0.0125241	0.530489	0.0249171	0.0105285
1.00	ITM	0.1680630	0.1862340	−0.01817040	10.81	0.00060843	0.0104021	0.457026	0.0117158	0.0119920
1.00	OTM	0.0632089	0.0585972	0.00461172	7.30	0.00044868	0.0202771	0.593674	0.0392519	0.0048317

Panel B: American put options

		Initial-time pricing				Execution-based L^2 errors				
T	Mon.	$V_0^{A,\mathrm{ref}}$	$P_{\mathrm{stop},0}$	$V_0^{A,\mathrm{ref}} - P_{\mathrm{stop},0}$	Rel. (%)	Price	Delta	Gamma	Vega	Theta
Offline learning										
0.25	ATM	0.0549432	0.0539498	0.00099339	1.81	0.00000213	0.0002033	45.5879	0.0001491	0.0010958
0.25	ITM	0.1194770	0.1208270	−0.00134996	1.13	0.00000500	0.0003898	31.9858	0.0006173	0.0012091
0.25	OTM	0.0187550	0.0185375	0.00021746	1.16	0.00000035	0.0001887	19.2842	0.0001420	0.0003543
0.50	ATM	0.0781519	0.0764059	0.00174593	2.23	0.00000915	0.0012862	169.2790	0.0008413	0.0021734
0.50	ITM	0.1395420	0.1383530	0.00118853	0.85	0.00000345	0.0008221	106.1190	0.0004430	0.0015167
0.50	OTM	0.0367646	0.0367660	−0.00000135	0.004	0.00000145	0.0011197	174.7650	0.0004093	0.0006767
1.00	ATM	0.1101540	0.1088660	0.00128856	1.17	0.00004216	0.0011300	119.6980	0.0012092	0.0004963
1.00	ITM	0.1700430	0.1681230	0.00192029	1.13	0.00000458	0.0016712	126.3410	0.0008602	0.0034781
1.00	OTM	0.0642271	0.0627601	0.00146694	2.28	0.00000147	0.0005592	80.9260	0.0004293	0.0005384
Online learning										
0.25	ATM	0.0549432	0.0543550	0.00058814	1.07	0.00000189	0.0003246	33.8512	0.0001415	0.0009268
0.25	ITM	0.1194770	0.1203610	−0.00088412	0.74	0.00000457	0.0004909	31.1525	0.0002280	0.0010911
0.25	OTM	0.0187550	0.0185775	0.00017747	0.95	0.00000047	0.0002312	19.4032	0.0000493	0.0004597
0.50	ATM	0.0781519	0.0770759	0.00107603	1.38	0.00000784	0.0006359	39.2434	0.0007039	0.0022082
0.50	ITM	0.1395420	0.1383810	0.00116097	0.83	0.00000282	0.0008366	59.0828	0.0003046	0.0019121
0.50	OTM	0.0367646	0.0366643	0.00010028	0.27	0.00000131	0.0003964	35.7629	0.0001948	0.0008532
1.00	ATM	0.1101540	0.1090340	0.00112077	1.02	0.00003452	0.0010904	19.8930	0.0010847	0.0008605
1.00	ITM	0.1700430	0.1675880	0.00245458	1.44	0.00000879	0.0004721	46.6444	0.0011499	0.0035762
1.00	OTM	0.0642271	0.0628865	0.00134058	2.09	0.00000156	0.0002508	60.1471	0.0003728	0.0004521

Notes. Panel A reports European put prices and grid-based price and Greek errors. Panel B reports American put prices and the corresponding execution-based grid errors. Mon. denotes moneyness: ITM, ATM, and OTM. In Panel A, Rel. is $|V_0^{E,*} - \widehat{V}_0^E|/V_0^{E,*}$; in Panel B, Rel. is $|V_0^{A,\mathrm{ref}} - P_{\mathrm{stop},0}|/V_0^{A,\mathrm{ref}}$. Both are reported as percentages, and the adjacent difference columns report signed pricing errors. For both panels, the Price and Greek L^2 errors are the mean squared errors over 61 equally spaced initial spot values in $[0.5, 1.5]$. The parameters are $S_0 = 1$, $r = 0.01$, $q = 0$, $\Delta t = 0.01$, and $T \in \{0.25, 0.50, 1.00\}$. The strikes are $K = 1.1$, $K = 1.0$, and $K = 0.9$ for ITM, ATM, and OTM puts, respectively. The evaluation Merton specification is $(\sigma, \eta_J, \mu_J, \delta_J) = (0.25, 0.50, -0.10, 0.20)$, with compensator $\kappa = -0.076884$. During European training, 8 Merton parameter specifications are sampled per iteration. During each American outer sweep, all nine contracts are updated using 4 sampled Merton parameter specifications per contract. The European stage is trained for 300 iterations and the American stage for 300 outer sweeps. Each parameter–contract training task uses 1024 paths. The American S_0 stopping prices use 10000 paths, while the American execution-surface and Greek errors use the same 2000 simulated shocks across the finite-difference perturbations. For the American stage, the penalty factor is $\Gamma = 10$, and the entropy temperature is annealed blockwise through $\lambda \in \{0.025, 0.005, 0.001\}$ over successive thirds of training.

Table 4.2 shows that the moderate-jump Merton experiment is more challenging for the European value surface than for the American stopping-based price. In Panel A, the mean European relative error is 16.97% for the offline learner and 12.24% for the online learner. The largest European errors are 28.98% for the offline $T = 0.25$ ATM contract and 29.27% for the online $T = 0.25$ OTM contract. The error is not uniform across maturity and moneyness: some ITM contracts remain relatively accurate, while several ATM and OTM contracts are materially less accurate. This variation is consistent with the greater difficulty of representing the parameter-conditioned Merton surface with one shared network.

The American stopping-based prices are more stable. In Panel B, the offline and online mean relative errors are 1.31% and 1.09%, respectively. The maximum errors are 2.28% for the offline $T = 1.00$ OTM contract and 2.09% for the corresponding online contract. Thus every American relative error is below 2.30%. The online method has the smaller absolute relative error in seven of the nine contracts, although the offline method is slightly better for the $T = 0.50$ OTM and $T = 1.00$ ITM contracts. The comparison therefore favours the online method on average, but not uniformly contract by contract. Fifteen of the eighteen signed errors are positive, indicating a mild tendency for the execution prices to lie below the dynamic-programming benchmark.

The relation between the European and American panels is not mechanical. For example, the offline European $T = 0.25$ ATM error is 28.98%, whereas the corresponding American stopping-based error is 1.81%. Similarly, the online European $T = 0.25$ OTM error is 29.27%, while the American error is 0.95%. The reported American value is the

average payoff obtained by executing the learned stopping rule; it is not calculated by adding the two network outputs at S_0. European approximation error therefore does not pass mechanically into the reported American price. At the same time, an accurate stopping price at S_0 does not by itself establish that the exercise boundary is equally accurate at every date and stock value.

The Greek results require a separate qualification. The American Price L^2 errors are small, and the Delta, Vega, and Theta errors remain on a much more controlled scale than the American Gamma errors. Gamma is computed by a second finite difference of the stopping-based price surface. The finite-sample execution surface can be nonsmooth because the deterministic rule changes the stopping date when the probability crosses 0.5. A small spot perturbation can therefore change the stopping date on some simulated paths, and the second difference amplifies both this change and the remaining Monte Carlo noise. The large American Gamma entries are consistent with this sensitivity of the deterministic stopping rule and its numerical differentiation; they do not contradict the accurate prices reported at S_0. They do, however, show that these learned execution prices should not be interpreted as providing a stable American Gamma estimate without additional smoothing or a larger Greek-evaluation sample.

Overall, Table 4.2 supports the robustness of the two-stage martingale-based method under the representative moderate finite-activity jump specification. The parameter-conditioned European Merton surface remains substantially harder to learn than the corresponding Black–Scholes surface. Nevertheless, with the Merton-specific state coverage and update organization, both learning modes produce accurate stopping-based American prices across

the nine reported contracts. The online procedure has the lower average error, while the large execution-based Gamma errors remain an important numerical limitation. Since the table evaluates one retained Merton specification and one trained run for each learning mode, the result should be interpreted as a robustness experiment rather than evidence of uniform accuracy over the full parameter pool.

4.3 Comparison with the Closed-Form European Baseline

The main American option pricing experiments in this thesis use the pre-trained European option price network as the baseline in the learned-baseline premium formulation. To compare this choice with the closed-form-baseline formulation used in Dai et al. (2025), we also run two baseline-comparison experiments, one using the offline procedure and one using the online procedure. In each experiment, the American option premium-learning component is trained in two versions. The first version uses the pre-trained European option price network as the baseline, while the second version uses the closed-form Black–Scholes European put value as the baseline.

Table 4.3 reports the last recorded iteration of this comparison in the offline experiment. The benchmark column $V_{\text{Bench},0}$ is the American reference price. The column $V_{\text{exec,learned},0}$ is the stopping-based American price obtained when the pre-trained European approximation is used as the baseline. The column $V_{\text{exec,closed},0}$ is the corresponding stopping-based price

obtained when the closed-form European value is used as the baseline.

Table 4.3: Offline comparison of the learned and closed-form European option price baselines in the American option premium-learning stage.

		American option benchmark	Learned European option price baseline		Closed-form European option price baseline	
T	Mon.	$V_{\text{Bench},0}$	$V_{\text{exec,learned},0}$	Rel. error (%)	$V_{\text{exec,closed},0}$	Rel. error (%)
0.25	ATM	0.058679	0.059176	0.85	0.060365	2.87
0.25	ITM	0.123381	0.122541	0.68	0.122647	0.59
0.25	OTM	0.019736	0.018573	5.89	0.019119	3.13
0.50	ATM	0.082307	0.078691	4.39	0.083158	1.03
0.50	ITM	0.144470	0.142242	1.54	0.144488	0.012
0.50	OTM	0.038682	0.038489	0.50	0.038090	1.53
1.00	ATM	0.115032	0.115338	0.27	0.114858	0.15
1.00	ITM	0.175920	0.173097	1.60	0.172209	2.11
1.00	OTM	0.067375	0.066881	0.73	0.066770	0.90

Notes. The table reports the offline American option baseline-comparison experiment after training iteration 500. The parameters are $S_0 = 1$, $r = 0.01$, $q = 0$, $\sigma = 0.30$, and $\Delta t = 0.01$. The maturities are $T \in \{0.25, 0.50, 1.00\}$. The strikes are $K = 1.1$, $K = 1.0$, and $K = 0.9$ for ITM, ATM, and OTM puts, respectively. The American option benchmark $V_{\text{Bench},0}$ is computed using the state-space dynamic-programming Bermudan approximation. In the learned European option price baseline version, the European option price network and the American option premium network are each trained for 500 iterations. In the closed-form European option price baseline version, only the American option premium network is trained, for 500 iterations. The training batch size is 2048, the evaluation batch size is 10000, the penalty factor is $\Gamma = 10$, and the entropy temperature is $\lambda = 1$. All nine contracts are included in each training iteration.

The comparison shows that the closed-form European baseline improves the average offline American stopping-price error, but the learned European baseline remains competitive. Across the nine contracts, the mean relative error is 1.83% with the learned baseline and 1.37% with the closed-form baseline. The closed-form baseline gives the smaller relative error in five of the nine contracts, which is expected because it removes the European baseline approximation error from the American option premium-learning procedure.

At the same time, the learned baseline performs better in four contracts: $T = 0.25$ ATM, $T = 0.50$ OTM, $T = 1.00$ ITM, and $T = 1.00$ OTM. This shows that using the analytic European value as the baseline does not automatically dominate the learned-

baseline construction contract by contract. Since the reported values are stopping-based execution prices, the final error depends on the stopping decisions induced by the learned premium surface, not only on the pointwise accuracy of the European baseline.

Overall, the closed-form baseline is advantageous when it is available, but the learned European option price baseline still produces stopping-based American option prices close to the selected benchmarks in this offline Black–Scholes comparison. This shows that the American option premium-learning stage can use a learned baseline instead of the available Black–Scholes formula. Because both asset-price models studied in this thesis admit analytic European benchmark formulas, the experiments do not directly establish performance in a setting where no such formula is available.

Table 4.4: Online comparison of the learned and closed-form European option price baselines in the American option premium-learning stage.

		American option benchmark	Learned European option price baseline		Closed-form European option price baseline	
T	Mon.	$V_{\text{Bench},0}$	$V_{\text{exec,learned},0}$	Rel. error (%)	$V_{\text{exec,closed},0}$	Rel. error (%)
0.25	ATM	0.058679	0.056701	3.37	0.059597	1.57
0.25	ITM	0.123381	0.120971	1.95	0.123155	0.18
0.25	OTM	0.019736	0.019527	1.06	0.020069	1.69
0.50	ATM	0.082307	0.081582	0.88	0.079684	3.19
0.50	ITM	0.144470	0.145583	0.77	0.140774	2.56
0.50	OTM	0.038682	0.038449	0.60	0.038733	0.13
1.00	ATM	0.115032	0.111031	3.48	0.115112	0.070
1.00	ITM	0.175920	0.176467	0.31	0.171326	2.61
1.00	OTM	0.067375	0.066482	1.33	0.066591	1.16

Notes. The table reports the online American option baseline-comparison experiment at epoch 500. The parameters are $S_0 = 1$, $r = 0.01$, $q = 0$, $\sigma = 0.30$, and $\Delta t = 0.01$. The maturities are $T \in \{0.25, 0.50, 1.00\}$. The strikes are $K = 1.1$, $K = 1.0$, and $K = 0.9$ for ITM, ATM, and OTM puts, respectively. The American option benchmark $V_{\text{Bench},0}$ is computed using the state-space dynamic-programming Bermudan approximation. In the learned European option price baseline version, the European option price network and the American option premium network are each trained for 500 iterations. In the closed-form European option price baseline version, only the American option premium network is trained, for 500 iterations. The training batch size is 2048, the evaluation batch size is 10000, the penalty factor is $\Gamma = 10$, and the entropy temperature is $\lambda = 1$. All nine contracts are included in each training iteration.

The online comparison gives a similar but more balanced picture. Across the nine contracts, the mean relative error is 1.53% with the pre-trained European baseline and 1.46% with the closed-form European baseline. Thus the closed-form baseline is slightly better on average, but the difference is small. The closed-form baseline gives the smaller error in five contracts, while the learned baseline gives the smaller error in four contracts: $T = 0.25$ OTM, $T = 0.50$ ATM, $T = 0.50$ ITM, and $T = 1.00$ ITM.

The largest online learned-baseline error is 3.48%, occurring for the $T = 1.00$ ATM contract, while the largest closed-form-baseline error is 3.19%, occurring for the $T = 0.50$ ATM contract. These results again show that the closed-form European baseline is useful when it is available, but it does not uniformly dominate the learned-baseline construction contract by contract. Since the reported values are stopping-based execution prices, the final error depends on the induced stopping rule rather than only on the pointwise accuracy of the European baseline.

4.4 Computational Cost

This section reports the computational cost of the Black–Scholes and moderate-jump Merton experiments. All runs were executed on Narval using 8 allocated CPU cores. Two types of runtime information are reported. Table 4.5 gives the wall-clock runtimes recorded by the training scripts and separates the European option price-learning component from the American option premium-learning component. Table 4.6 reports the corresponding Slurm accounting summaries for the complete jobs.

The Merton entries correspond to the final implementation with 300 European training iterations and 300 American outer sweeps, where one outer sweep means one complete pass through all nine contracts. The script-recorded totals include small costs outside the two separately timed components and may therefore differ slightly from their sum. Some environment and process startup, common post-training evaluation, packaging, and file-saving operations also occur outside the timed regions and are reflected only in the Slurm elapsed time.

Table 4.5: Wall-clock runtimes for the Black–Scholes and moderate-jump Merton experiments on Narval.

	Runtime (d:hh:mm:ss)		
Learning mode	European option price stage	American option premium stage	Total
Black–Scholes dynamics			
Offline learning	0:01:05:00	0:07:03:17	0:08:08:17
Online learning	0:06:00:08	0:13:57:39	0:19:57:47
Moderate-jump Merton dynamics			
Offline learning	0:00:02:42	0:01:52:10	0:01:54:52
Online learning	0:00:04:02	0:02:15:52	0:02:19:54

Notes. In each learning mode, the Black–Scholes experiment uses 1000 European option price training iterations and 1000 American option premium training iterations, with 2048 paths per batch. The Merton experiment uses 300 European option price training iterations, with 8 batches per iteration, and 300 American option premium outer sweeps, with 36 batches per outer sweep and 1024 paths per batch. The reported runtimes are rounded to the nearest second.

The American premium-learning component is the dominant computational component under both asset-price dynamics and both training modes. In the Black–Scholes offline run, it accounts for about 86.7% of the total runtime, while in the Black–Scholes online run it accounts for about 69.9%. Its share is even larger in the final Merton experiment: the American component accounts for about 97.6% of the offline runtime and 97.1% of the online runtime.

The organization of the optimizer updates is important for interpreting these runtimes.

An optimizer update is one application of the calculated gradient to the network parameters. In the Black–Scholes offline implementation, the losses from all nine contracts are combined before one optimizer update is applied in each iteration. Therefore, 1000 iterations produce 1000 optimizer updates in each of the European and American stages.

The Black–Scholes online implementation applies the optimizer more frequently. For each contract, an update is applied immediately after calculating the one-step CTD(0) residual at each decision date. There are three contracts at each of 25, 50, and 100 decision steps. Consequently, one online iteration produces $3(25+50+100) = 525$ optimizer updates. Over 1000 iterations, this gives 525,000 updates in each of the European and American stages. Each of these online updates uses only one contract and one pair of adjacent dates and is therefore much smaller than a full offline update. The numbers of optimizer updates should consequently not be interpreted as numbers of operations with equal computational cost.

The final Merton implementation uses a different update organization. In the European stage, each iteration samples 8 Merton specifications and pairs them with contracts. For each parameter–contract pair, the loss or one-step objective is evaluated at 8 selected dates. The resulting gradients are averaged before one optimizer update is applied. Thus, both the offline and online European procedures apply one optimizer update per iteration, giving 300 updates in each mode.

In the American stage, one outer sweep passes once through all nine contracts. For each contract, gradients from 4 sampled Merton specifications and 8 selected dates are

combined before one optimizer update is applied. Each outer sweep therefore involves 36 parameter–contract gradient calculations but only 9 optimizer updates. Over 300 outer sweeps, this produces $300 \times 9 = 2700$ American optimizer updates in each learning mode.

Combining the online residuals before applying an optimizer update does not change the online procedure into an offline procedure. In the online mode, each residual still uses information from two adjacent dates. In the offline mode, each residual uses the full future return from its selected starting date. The distinction between the two methods is therefore determined by the information used to construct the learning signal, rather than by whether every individual residual is followed immediately by an optimizer update.

The online procedure remains slower than the offline procedure under both dynamics. Under Black–Scholes dynamics, the total online runtime is about 2.45 times the total offline runtime. This comparison reflects both the different loss calculations and the large number of sequential optimizer updates in the Black–Scholes online implementation. Under the moderate-jump Merton dynamics, the online runtime is about 1.22 times the offline runtime, or approximately 21.8% longer. In this case, the two modes use the same optimizer-update schedule: one European update per iteration and nine American updates per outer sweep.

The remaining Merton runtime difference is therefore associated with the calculations required by the two learning objectives. The offline method calculates a full future return from each selected date. The online method evaluates learned values at adjacent dates to form its one-step residuals. In the American option premium-learning stage, the online

calculation also evaluates the frozen European option price network at both adjacent dates to correct for any remaining one-step martingale error in the learned European option price baseline, as described in Section 4.2. The recorded timings do not isolate the individual contribution of each of these calculations.

The final Merton runtimes are lower than the corresponding Black–Scholes runtimes reported in Table 4.5. This should not be interpreted as evidence that jump–diffusion simulation is intrinsically less expensive than Black–Scholes simulation. The two experiments use substantially different numerical organizations. The Black–Scholes implementation uses 1000 iterations and, in the online mode, applies an update at every decision date. The final Merton implementation uses 300 European iterations and 300 American outer sweeps, evaluates each training task at 8 selected dates, and combines gradients before updating the network.

As discussed in Section 4.2, directly transferring the Black–Scholes numerical strategy did not provide sufficient balanced coverage of the contracts and exercise-relevant states under the Merton model and produced stopping policies that continued most paths to maturity. The final Merton implementation therefore uses adapted state sampling and a different organization of the optimization updates. The underlying offline full-return and online one-step methodologies remain unchanged. The reported runtimes compare the complete implementations used for the two experiments rather than providing a controlled comparison of the intrinsic costs of diffusion and jump–diffusion simulation.

Slurm accounting provides additional information about CPU and memory usage.

Elapsed time is the total wall-clock duration of a job, while total CPU time is the processing time accumulated across the job's tasks. Peak memory is the largest memory usage recorded for the batch step. In both experiments, the offline and online procedures were executed sequentially within one complete job. Therefore, the Slurm measurements in Table 4.6 are available only for the combined jobs and cannot be separated by learning mode.

Table 4.6: Slurm resource summary for the complete Black–Scholes and moderate-jump Merton jobs on Narval.

Asset-price dynamics	Allocated CPUs	Time (d:hh:mm:ss)		Peak memory (GB)
		Elapsed	Total CPU	
Black–Scholes	8	1:04:08:00	1:22:40:00	10.1
Moderate-jump Merton	8	0:04:22:00	0:05:05:00	2.5

Notes. The resource summary refers to the complete computational jobs containing both the offline and online learning modes. The corresponding training configurations are reported in Table 4.5.

The final Merton job used approximately 4.37 hours of elapsed wall-clock time and reached a peak memory usage of about 2.5 GB. The offline and online script-recorded totals sum to approximately 4.25 hours. The difference of about 7 minutes includes environment and process startup and other operations outside the timed regions, such as the common post-training evaluation and packaging. The Slurm accounting output does not allow this difference to be separated precisely among these operations. Similarly, the combined Black–Scholes script runtimes are close to the elapsed time reported for the complete Black–Scholes job.

The final Merton job used less peak memory and elapsed time than the complete Black–Scholes job. However, as with the script-recorded runtimes, these measurements should not be used to conclude that the Merton model is generally less computationally demanding.

They reflect different training schedules, state-sampling procedures, and organizations of the optimizer updates. Moreover, because the two learning modes were executed within the same job, the Slurm output does not support a separate comparison of their peak memory requirements.

For both experiments, total CPU time is considerably smaller than the product of the elapsed time and the 8 allocated CPU cores. This indicates that all allocated cores were not fully used throughout training. This pattern is consistent with sequential calculations along the exercise grid, repeated network evaluations, and Python-level loops that were not fully parallelized in the present implementation. The Slurm accounting output alone, however, does not identify the precise contribution of each of these operations.

Overall, the adapted Merton implementation completes each learning mode within a few hours, with the offline procedure remaining faster than the online procedure. Across both asset-price models, the American premium-learning stage is the main source of computational cost. Future implementations would therefore benefit from more efficient batching across contracts and Merton specifications, fewer repeated network evaluations during the policy calculations, and stronger vectorization along the exercise grid.

4.5 Summary and Interpretation

Table 4.7 summarizes the relative pricing errors at S_0 reported in Tables 4.1 and 4.2. The mean relative error is computed across the nine maturity–moneyness contracts in each

group, and the maximum relative error records the largest contract-level error.

Table 4.7: Summary of initial-time relative pricing errors by asset-price dynamics, learning mode, and option type.

Option type	Mean relative error (%)	Maximum relative error (%)
Black–Scholes dynamics		
Offline learning		
European put options	4.57	8.90
American put options	1.34	2.40
Online learning		
European put options	3.96	8.01
American put options	1.03	2.97
Moderate-jump Merton dynamics		
Offline learning		
European put options	16.97	28.98
American put options	1.31	2.28
Online learning		
European put options	12.24	29.27
American put options	1.09	2.09

Notes. The mean and maximum are computed across the nine maturity–moneyness contracts for each asset-price dynamics, learning mode, and option type. The values are the initial-time relative pricing errors reported in Tables 4.1 and 4.2.

The first pattern is that the reported American execution-price errors are substantially smaller than the direct European network-value errors. Under Black–Scholes dynamics, the mean relative error decreases from 4.57% to 1.34% for the offline method and from 3.96% to 1.03% for the online method. Under the moderate-jump Merton dynamics, the corresponding means decrease from 16.97% to 1.31% offline and from 12.24% to 1.09% online.

These comparisons assess different outputs, however. The European price is obtained by evaluating the learned European value surface directly at the initial state. The reported American price is instead a Monte Carlo estimate of the discounted payoff obtained by executing the deterministic stopping rule induced by the learned American component. It is not obtained by simply adding the European and American network outputs at S_0.

Consequently, European value-surface error does not pass mechanically into the reported American execution price.

The second pattern is the substantially larger European error in the Merton experiment. The mean errors are 16.97% offline and 12.24% online, while the maximum errors are 28.98% for the offline $T = 0.25$ ATM contract and 29.27% for the online $T = 0.25$ OTM contract. These results are consistent with the greater difficulty of representing a shared contract- and parameter-conditioned surface over the retained family of Merton specifications. The increase cannot be attributed to jumps alone, however, because the Black–Scholes and Merton experiments use different network inputs, normalization choices, state-sampling procedures, and optimization schedules.

Despite the larger European errors, the final Merton stopping-based prices remain close to the dynamic-programming benchmarks. The offline and online mean American errors are 1.31% and 1.09%, respectively. Their maximum errors are 2.28% and 2.09%, both for the $T = 1.00$ OTM contract. Thus, every reported Merton American execution-price error is below 2.30%. The online method has the smaller error in seven of the nine American contracts, although the offline method is slightly more accurate for the $T = 0.50$ OTM and $T = 1.00$ ITM contracts.

The online learning procedure has the smaller mean relative pricing error in each of the four reported comparisons, but the contract-level results do not show uniform superiority. Across the 36 contract-level comparisons, the online learning procedure has the smaller reported relative error in 25 cases, while the offline learning procedure has the smaller

reported relative error in 11 cases. The results are most evenly divided for the Black–Scholes European put options, for which the online learning procedure has the smaller error for four of the nine contracts and the offline learning procedure has the smaller error for five. The online learning procedure has the smaller error for six of the nine Black–Scholes American put options, eight of the nine Merton European put options, and seven of the nine Merton American put options. The exceptions are nevertheless important. Under Black–Scholes dynamics, the online $T = 0.25$ OTM American put option has a relative pricing error of 2.97%, compared with 1.27% offline, and this error exceeds the offline American maximum of 2.40%.

Under the moderate-jump Merton jump–diffusion dynamics, the online $T = 0.25$ OTM European put option has a relative pricing error of 29.27%, compared with 0.44% offline. This single contract makes the maximum online European error slightly larger than the corresponding offline maximum of 28.98%. Thus, the online learning procedure is more accurate on average and for most individual contracts, particularly in the moderate-jump Merton jump–diffusion experiment, but isolated contract-specific errors prevent a conclusion of uniform superiority.

The organization of the optimizer updates also differs between the two asset-price implementations, as discussed in Section 4.4. In the Black–Scholes experiment, the online method applies sequential updates along the decision grid and therefore receives many more optimizer updates than the offline method. In the final Merton implementation, by contrast, the one-step online residuals are combined before an update is applied. The offline and online Merton procedures use the same optimizer-update schedule: one European

update per iteration and nine American updates per outer sweep. The lower online Merton mean errors therefore cannot be attributed to a larger number of optimizer updates. They are descriptive outcomes of the different full-return and one-step learning objectives and of the particular trained runs.

The American price errors also provide only indirect information about the accuracy of the stopping boundary. At the representative Merton specification, the benchmark early-exercise premium is only about 0.5% to 1.8% of the total American value across the nine contracts. A policy can therefore produce a small total-price error while recovering only part of the early-exercise premium. Accurate execution prices at S_0 do not by themselves establish uniform accuracy of the exercise boundary, the value surface, or the associated Greeks.

Overall, the summary statistics support the pricing conclusion of the numerical study in a qualified form. Under Black–Scholes dynamics, both learning modes produce American execution prices with mean errors close to 1% and maximum errors below 3%. Under the representative moderate-jump Merton specification, the adapted state coverage, temperature scale, and update organization produce mean American errors close to 1% and maximum errors below 2.30%, despite the larger direct European value-surface errors. These results provide evidence that the two-stage construction can generate accurate execution prices after the introduction of finite-activity jumps. They do not establish uniform accuracy over the complete Merton parameter pool or uniform recovery of the optimal stopping boundary.

5

Conclusion and Future Work

5.1 Summary of Findings

This thesis developed a two-stage martingale-based learning framework for pricing European and American put options. The method first pre-trains a European value approximation from simulated risk-neutral paths. This approximation is then held fixed and used as

the baseline for learning an American early-exercise premium. More precisely, the American network learns a correction relative to the learned European baseline. With an exact European baseline, this correction targets the economic early-exercise premium $V_A - V_E$, although finite penalty, entropy regularization, time discretization, and approximation errors mean that the learned correction need not coincide with it exactly.

Overall, the numerical results support the main premise of the thesis: the learned deterministic stopping rule can produce accurate American execution prices even when the direct European network estimate at S_0 is less accurate. Both learning modes perform consistently under Black–Scholes dynamics, while the adapted Merton implementation learns meaningful early-exercise regions and produces mean American relative errors of 1.31% offline and 1.09% online, with all nine reported Merton American relative pricing errors below 2.30% for both modes. Since these American prices are Monte Carlo execution values obtained from the learned stopping rule, rather than direct evaluations of the reconstructed American value $\widehat{V}_E + U_{A,\theta_A}$ at S_0, European baseline errors need not pass one-for-one into them. Nevertheless, accurate execution prices at S_0 do not establish uniform accuracy of the exercise boundary, reconstructed value surface, or associated Greeks.

The experiments also show that the offline procedure is faster under both asset-price dynamics. Under Black–Scholes dynamics, the online procedure updates the network sequentially along the decision grid, contributing to its longer runtime. In the final Merton experiment, however, the procedure labelled online combines several one-step CTD(0) residuals before each contract-level update. Both Merton procedures therefore follow the same optimizer-update schedule: one European option price-network update per iteration

and nine American option premium-network updates per outer sweep. Thus, the Merton labels distinguish the full-return and one-step objectives, rather than whether the network is updated after every decision date. The one-step procedure requires approximately 2.33 hours, compared with 1.91 hours for the full-return procedure. This difference cannot be explained by the number of updates and is consistent with the different computations required by the two objectives, although the precise source cannot be isolated from the aggregate timings. Since Slurm reports resources only for the combined Merton job, their memory requirements cannot be compared separately.

5.2 Limitations

Several limitations remain. First, the numerical study of European and American option pricing does not include a market-calibration stage. All numerical experiments use simulated risk-neutral stock-price paths under prescribed risk-neutral asset-price model parameters. This is appropriate for controlled numerical validation of the option-pricing framework, but it does not establish the empirical option-pricing performance of the framework after the risk-neutral asset-price model is calibrated to an observed panel of market option prices. In the present implementation, the contract- and parameter-conditioned European option value network and the stopping-based American option pricing procedure are used only in the forward direction: the risk-neutral asset-price model parameters and option-contract variables are inputs, and the corresponding option prices are outputs. Market calibration requires the inverse task of recovering the common risk-neutral asset-price

model parameters from observed option prices.

This inverse calibration problem could be approached in two ways. First, the learned forward pricing map could be evaluated repeatedly within a constrained optimization procedure that minimizes a weighted discrepancy between observed market option prices and model-implied option prices. Second, a separate inverse neural network could be trained on simulated pairs of risk-neutral asset-price model parameters and option-price panels, with the option-price panel as the input and the corresponding model parameters as the output. However, the inverse map is not obtained simply by reversing the trained pricing networks. In a model with several unknown parameters, a single option price generally does not identify all of the parameters. The inverse input would therefore need to contain a sufficiently informative panel of European and American-style option prices across different strikes and maturities. Even with such a panel, different parameter combinations may generate similar option prices, making the inverse calibration problem non-unique or numerically ill-conditioned. A market-calibrated implementation would therefore need to investigate parameter identifiability, parameter constraints, regularization, sensitivity to market noise, and calibration uncertainty. The present thesis implements neither the optimization-based inversion of the learned forward pricing map nor a separately learned inverse calibration map. This is a limitation of the present numerical option-pricing experiments rather than a restriction preventing the two-stage martingale-based option-pricing framework from incorporating market information.

A second limitation concerns the reuse, within each learning mode, of one pre-trained European value surface as a common baseline across the option-contract panel. A sys-

tematic baseline error can therefore affect the shared American option premium-learning procedure and the induced stopping rules for several contracts simultaneously. Such an error need not transfer one-for-one into the reported American prices: those prices are Monte Carlo averages of discounted payoffs obtained by executing the learned deterministic stopping rule, rather than evaluations of the reconstructed American value $\widehat{V}_E + U_{A,\theta_A}$ at S_0. Consequently, European baseline error can affect the reported price indirectly through its influence on the learned exercise decisions.

The final Merton implementation includes the frozen-European martingale defect in both American learning objectives. The full-return objective adds the accumulated future defect to its residual, whereas the batched one-step CTD(0) objective adds the corresponding one-step defect. These corrections account for the component of the sampled training residual arising from an imperfect frozen baseline, but they do not make the European network exact or eliminate neural-network approximation, finite-sample, and optimization errors. Complete compensation for a systematic baseline error is therefore not guaranteed, and remaining error can affect the exercise comparison, particularly near the stopping boundary.

The comparison using a closed-form Black–Scholes baseline and the final Merton results indicates that baseline error need not pass mechanically into the stopping-based American execution price. However, these experiments do not isolate error propagation across the contract family or establish uniform accuracy of the reconstructed American value surface, the exercise boundary, or the execution-based Greeks. Robustness outside the parameter region used in training or under changing market conditions is also not established.

Both asset-price models examined in this thesis admit analytic European pricing formulas: the Black–Scholes formula and the Merton series representation. Apart from the explicit closed-form-baseline comparison in Section 4.3, these formulas do not enter the training procedure and are used only to construct evaluation benchmarks. Accordingly, the experiments demonstrate that the implemented training procedure can omit an available analytic European formula. They do not directly evaluate a model for which no such formula exists.

Third, the Merton extension is restricted to a curated moderate-jump pool. The log-jump sizes are normally distributed, jump activity is finite, and the retained parameter range avoids extreme jump behaviour. The final implementation also relies on Merton-specific numerical adaptations, including broader starting-state coverage, greater representation of downside and exercise-relevant states, balanced updates across contracts and sampled specifications, a scaled decision-gap parameterization, and an annealed entropy temperature. These adaptations were motivated by preliminary runs in which the learned policies continued most paths to maturity, but their separate contributions are not isolated through an ablation study.

The final table evaluates one representative specification from the retained training pool on independent paths; it is not an out-of-pool parameter test. Moreover, only one trained run is reported for each learning mode, so training-seed variability is not measured. The results therefore do not establish uniform pricing or boundary accuracy across the retained parameter pool. Stronger jump intensities, alternative state-sampling designs, alternative jump-size distributions, time-varying jump parameters, and infinite-activity models are not

studied.

Fourth, volatility dynamics remain limited. In both the Black–Scholes and Merton experiments, the diffusive volatility is treated as a conditioning parameter rather than as a dynamic state variable. The thesis does not consider stochastic volatility or conditional heteroskedasticity. This excludes models such as GARCH, in which conditional variance evolves with past squared innovations and past conditional variances. The GARCH option-pricing model of Duan (1995) is a natural reference point for this type of extension.

Fifth, the numerical study is one-dimensional and focuses on put options. This is a natural first test case because, under the parameter settings considered, American puts exhibit nontrivial early-exercise behaviour and reliable numerical benchmarks can be computed. However, the experiments do not address basket options, contracts written on multiple correlated assets, path-dependent payoffs, or more complex exercise features.

Finally, the reported option Greek errors measure the local accuracy of the estimated option price sensitivities rather than the performance of a self-financing dynamic hedging strategy. For American put options, the execution-based option price sensitivities are obtained by recomputing the stopping-based American option price under small perturbations of the current stock price, the diffusion volatility parameter, and the remaining time to maturity, using the same trained American option premium network and the deterministic stopping rule induced by that network. Although the estimated American option delta could in principle be used as the hedge ratio for a position in the underlying stock, the thesis does not evaluate a self-financing dynamic hedging strategy along out-of-sample

stock-price paths. The numerical results therefore do not establish whether the estimated American option price sensitivities reduce realized dynamic hedging error.

Higher-order American option price sensitivities also remain difficult to estimate accurately. In particular, the estimated American option gamma is less stable than the American option price and the estimated delta, vega, and theta because gamma is a second derivative of the American option price with respect to the current stock price and is strongly affected by local nonsmoothness near the put option payoff kink and the learned American option early-exercise boundary. A practical smoothing solution would be to evaluate the stopping-based American option price on a dense local grid of initial stock prices using common random numbers and then fit a penalized cubic smoothing spline to the resulting execution-based American option price curve. The American option delta and gamma could then be obtained from the first and second derivatives of the smoothed price curve rather than from direct finite differences of individual Monte Carlo price estimates whose stopping decisions may change under small stock-price perturbations. The smoothing parameter would need to be selected using held-out simulated paths or a sensitivity analysis, because excessive smoothing could reduce variance at the cost of introducing bias near the put option payoff kink and the American option early-exercise boundary. This smoothing procedure would be applied only during the post-training estimation of the American option Greeks. It would not alter the deterministic American option stopping rule or the stopping-based American option prices reported in the numerical experiments.

5.3 Directions for Future Work

A natural first direction is to train the European option's time-value correction and the American option premium within the same overall training loop, rather than first training and freezing the European option price network. Comparing this joint training procedure with the present two-stage construction would show whether an evolving European option price baseline improves the learning of the American option premium or introduces instability through the simultaneous updating of the two neural networks. A related convergence and error-propagation analysis would examine how the penalty factor, the entropy-temperature parameter, the time-step size, optimization error, neural-network approximation error, European option price baseline approximation error, and the choice between the full-return and one-step CTD(0) learning procedures affect the learned deterministic American option stopping rule and the final stopping-based American option price.

This numerical error analysis should be complemented by an analysis of structural model risk. In particular, model misspecification may arise when the assumed risk-neutral asset-price dynamics, state representation, or volatility and jump specifications do not adequately represent the observed option-price surface. Regime changes may also shift volatility, jump behaviour, or other market conditions outside the asset-price model parameter region represented during training or calibration. These risks cannot be resolved merely by increasing the number of simulated paths, because additional paths reduce Monte Carlo

estimation uncertainty under the assumed model but do not correct an incorrect model specification or an unrepresented market regime. Future work should therefore stress-test the shared European option price baseline and the American option premium-learning component under alternative asset-price models and changing market regimes, evaluate performance outside the training parameter region, and establish criteria for recalibration or retraining when the observed option-price surface changes materially.

A second direction is to broaden the Merton jump–diffusion numerical experiments at two levels. Within the standard Merton jump–diffusion model, future work could evaluate rare-but-severe, frequent-but-small, and more extreme downside-jump parameter regimes. These experiments would provide stress tests across different parameter specifications within the same Merton jump–diffusion model class. Changing the Merton jump–diffusion model parameters does not change the structure of the infinitesimal generator. It changes only the parameter values appearing in the diffusion, drift-compensation, and jump terms. Therefore, there is no structural reason that the martingale-based learning framework should cease to apply under a different valid Merton parameter specification. However, the numerical accuracy of the learned European option value surface, the American option premium, and the deterministic American option stopping rule outside the curated moderate-jump training region would need to be investigated separately.

A separate extension would change the log-jump-size distribution, for example by replacing Merton's Gaussian log-jump-size distribution with the double-exponential distribution (Kou, 2002) or the mixed-exponential distribution (Cai and Kou, 2011). For the continuous-time Markov asset-price models considered here, the corresponding infinitesi-

mal generator can be obtained through the same general three-step procedure. First, the risk-neutral asset-price dynamics and all state variables required to make the asset-price model Markov are specified. Second, Itô's formula, or its jump-process extension, is applied to a sufficiently smooth function of the state variables. Third, the predictable terms proportional to dt are collected, and these terms define the infinitesimal generator. For the double-exponential and mixed-exponential jump–diffusion models, the generator retains the same general one-dimensional structure as the Merton generator, but the log-jump-size distribution and the corresponding jump compensator are changed.

More ambitious extensions could consider stochastic-volatility jump–diffusion models (Bates, 1996) or infinite-activity Lévy asset-price dynamics such as the CGMY model (Carr et al., 2002). The Bates model also has an infinitesimal generator, but the instantaneous variance must be included as an additional Markov state variable, so the generator becomes higher-dimensional. The CGMY model also has an infinitesimal generator, but its infinite jump activity produces a more complicated nonlocal jump term and requires a suitable infinite-activity path simulator. The same simple continuous-time finite-dimensional generator construction does not apply directly to a discrete-time model such as GARCH, which is described by a one-step conditional transition rule, or to a genuinely non-Markovian asset-price model without first enlarging the state representation or using a path-dependent formulation. These extensions would therefore require model-specific changes to the risk-neutral asset-price path simulator, the neural-network inputs, and the option-pricing benchmark procedure. They would test whether the observed option-pricing performance depends on the finite-activity Gaussian log-jump-size assumptions used in the

present numerical study.

A third direction is to introduce dynamic volatility through the Heston stochastic-volatility model (Heston, 1993). Under the Heston model, the stock price and the instantaneous variance form a two-dimensional continuous-time Markov state. The state representation of the European option value network and the American option premium network would therefore include the current instantaneous variance in addition to the current stock price. The corresponding infinitesimal generator can be obtained by applying Itô's formula to a sufficiently smooth function of time, stock price, and instantaneous variance and then collecting the predictable terms proportional to dt. This makes the Heston stochastic-volatility model a direct extension of the continuous-time martingale-based policy-evaluation framework used in this thesis.

The Heston–Nandi GARCH model provides a related discrete-time alternative (Heston and Nandi, 2000). Its single-lag specification has the Heston stochastic-volatility model as a continuous-time diffusion limit. However, the original Heston–Nandi GARCH model remains discrete in time and would be formulated using a one-step risk-neutral transition operator and discrete-time Bellman or martingale relations rather than the continuous-time infinitesimal generator used in the present thesis. A direct Heston extension would therefore provide the more natural first test of whether the learned European option price baseline, the American option premium, and the deterministic American option stopping rule remain accurate when volatility is an additional dynamic state variable.

A fourth direction is to evaluate the practical usefulness of the estimated American

option delta through a simulated self-financing dynamic hedging experiment. On an independent set of out-of-sample stock-price paths, a short position in an American put option could be hedged by rebalancing the position in the underlying stock on the American option exercise grid using the execution-based American option delta estimated from the deterministic stopping rule induced by the learned American option premium network. The residual cash position would be invested in, or borrowed from, the risk-free account until early exercise or maturity. To distinguish the dynamic hedging error arising from the execution-based American option delta estimate from the error arising from the learned deterministic American option stopping rule, one controlled comparison could terminate both the hedging strategy based on the learned American option delta and the hedging strategy based on the benchmark American option delta under the same benchmark American option exercise rule. A second, end-to-end comparison could allow the hedging strategy based on the learned American option delta to use the deterministic American option exercise rule induced by the learned American option premium network. The resulting distributions of the dynamic hedging error at the American option exercise time or maturity could then be compared with each other and with the hedging-error distribution of an unhedged short American put option position.

Under Black–Scholes dynamics, this experiment would test whether the pointwise delta accuracy reported in Chapter 4 translates into approximate replication performance under discrete rebalancing. Under Merton jump–diffusion dynamics, the experiment would instead measure risk reduction, since stock-and-cash delta hedging cannot eliminate jump risk. This distinction is consistent with Gao, L. Li, and Zhou (2026), who formulate a

mean–variance option-hedging problem under jump–diffusion dynamics. Their optimal risky-asset position combines an option-delta component for continuous price movements with adjustments for jump risk and for the discrepancy between the hedging-portfolio value and the option value. They evaluate the learned policy using out-of-sample mean-squared hedging error. Although their application concerns European-style options, it provides a useful template for extending the present American-option framework from pointwise Greek estimation to dynamic hedging.

A fifth direction is to develop a market-calibrated implementation of the two-stage martingale-based option-pricing framework. The common risk-neutral asset-price model parameters could be recovered from a panel of European and American-style option quotes either by repeatedly evaluating the learned forward pricing map within a constrained optimization procedure or by training a separate inverse network that maps an informative option-price panel to model-parameter estimates. Both approaches would need to address parameter identifiability, market noise, and calibration uncertainty.

After calibration, the hierarchical construction could identify several option-price components. The difference between the calibrated European option value and the contractual payoff would measure the European option's time-value correction, while the difference between the stopping-based American option price and the calibrated European option value would provide an execution-based estimate of the early-exercise premium. Counterfactual repricing could also measure model-implied jump or volatility contributions, for example by comparing the calibrated Merton option price with the price obtained after removing the jump component. These quantities are risk-neutral option-price decompositions, not

economic risk premia. Identifying variance or jump risk premia would additionally require estimating the corresponding asset-price dynamics under the physical probability measure from historical returns and comparing them with the parameters calibrated under the risk-neutral measure. The present thesis does not implement these calibration, decomposition, or risk-premium estimation procedures.

Finally, the method could be extended to higher-dimensional and path-dependent American-style derivatives, including basket options and contracts written on multiple correlated assets. Multi-asset contracts would require an enlarged state representation, while path-dependent contracts could require suitable state augmentation or recurrent architectures. Such problems are difficult for classical full-grid PDE methods because of the curse of dimensionality, which is one reason learning-based methods have become attractive for high-dimensional PDEs and control problems (Han, Jentzen, and E, 2018; Huré et al., 2021). Extending the present framework to these settings would provide a direct test of its scalability, which is not established by the one-dimensional experiments in this thesis.

A

Black–Scholes Benchmark Greeks

This appendix chapter records the Black–Scholes benchmark Greeks used in the numerical experiments. These formulas are obtained by differentiating the European put price (2.10).

The delta of the European put is

$$\Delta_E(t,s) := \frac{\partial V_E}{\partial s}(t,s) = -e^{-q(T-t)}\Phi(-d_1(t,s)). \tag{A.1}$$

It measures the first-order sensitivity of the European put value with respect to the stock price.

The gamma

$$\Gamma_E(t,s) := \frac{\partial^2 V_E}{\partial s^2}(t,s) = \frac{e^{-q(T-t)}\phi(d_1(t,s))}{s\sigma\sqrt{T-t}}, \tag{A.2}$$

measures the second-order sensitivity of the European put value with respect to the stock price.

The vega

$$\text{Vega}_E(t,s) := \frac{\partial V_E}{\partial \sigma}(t,s) = se^{-q(T-t)}\phi(d_1(t,s))\sqrt{T-t}, \tag{A.3}$$

measures the sensitivity of the European put value with respect to the volatility parameter σ.

Finally, theta is defined as the derivative with respect to the current time variable t. With this convention,

$$\Theta_E(t,s) := \frac{\partial V_E}{\partial t}(t,s) = -\frac{s\sigma e^{-q(T-t)}\phi(d_1(t,s))}{2\sqrt{T-t}} + rKe^{-r(T-t)}\Phi(-d_2(t,s)) - qse^{-q(T-t)}\Phi(-d_1(t,s)), \tag{A.4}$$

measures the sensitivity of the European put value with respect to the current time variable. This sign convention is consistent with the pricing equation (2.7). Since $T-t$ is the remaining time to maturity, this definition is the negative of the derivative with respect to time to maturity.

For $t < T$, the Black–Scholes value is smooth. At maturity, $V_E(T,s) = g_K(s)$ for $s > 0$.

B

Supplementary Derivations

B.1 Offline European Regression Loss

This appendix section gives the conditional-expectation argument behind the offline European regression loss used in Section 3.1. It also explains how this implemented regression loss is related to the exact martingale-loss objective.

We have already defined the exact discounted martingale loss:

$$\mathcal{L}_E^{\mathrm{ML}}(\theta) := \frac{1}{2}\,\mathbb{E}^{\mathbb{Q}}\left[\int_0^T \left|e^{-rT} g_K(S_T) - e^{-rt} V_{E,\theta}(t, S_t)\right|^2 dt\right]. \tag{B.1}$$

To interpret this objective, define the discounted mean-square value error by

$$\mathrm{DMSVE}_E(\theta) := \mathbb{E}^{\mathbb{Q}}\left[\int_0^T e^{-2rt} \left|V_E(t, S_t) - V_{E,\theta}(t, S_t)\right|^2 dt\right]. \tag{B.2}$$

This quantity measures the discounted mean-square distance between the true European value and its approximation along simulated paths.

The following result is a direct specialization of Theorem 3 in Jia and Zhou (2022) to the discounted European option value process.

Theorem B.1 (Exact martingale loss and discounted value error)**.** *The following identity is a direct specialization of the martingale-loss decomposition of Jia and Zhou (2022) to the discounted European option value process. The exact martingale loss* $\mathcal{L}_E^{\mathrm{ML}}(\theta)$ *and the discounted mean-square value error* $\mathrm{DMSVE}_E(\theta)$ *have the same minimizers. More precisely,*

$$\mathcal{L}_E^{\mathrm{ML}}(\theta) = C_E^{\mathrm{ML}} + \frac{1}{2}\,\mathrm{DMSVE}_E(\theta), \tag{B.3}$$

where

$$C_E^{\mathrm{ML}} = \frac{1}{2}\,\mathbb{E}^{\mathbb{Q}}\left[\int_0^T \left|e^{-rT} g_K(S_T) - e^{-rt} V_E(t, S_t)\right|^2 dt\right],$$

is independent of θ*, so it does not affect the minimization problem.*

Proof. For each fixed $t \in [0,T]$, the martingale property of the discounted European value process gives

$$e^{-rt}V_E(t,S_t) = \mathbb{E}^{\mathbb{Q}}\left[e^{-rT}g_K(S_T) \mid \mathcal{F}_t\right].$$

Hence $e^{-rt}V_E(t,S_t)$ is the conditional expectation of $e^{-rT}g_K(S_T)$ given $\mathcal{F}_t$. Since $e^{-rt}V_{E,\theta}(t,S_t)$ is also $\mathcal{F}_t$-measurable, the projection property of conditional expectation yields

$$\begin{aligned}\mathbb{E}^{\mathbb{Q}}\Big[|e^{-rT}g_K(S_T) - e^{-rt}V_{E,\theta}(t,S_t)|^2\Big] &= \mathbb{E}^{\mathbb{Q}}\Big[|e^{-rT}g_K(S_T) - e^{-rt}V_E(t,S_t)|^2\Big] \\ &\quad + \mathbb{E}^{\mathbb{Q}}\Big[e^{-2rt}|V_E(t,S_t) - V_{E,\theta}(t,S_t)|^2\Big].\end{aligned}$$

Integrating this identity over $t \in [0,T]$ and multiplying by $1/2$ gives (B.3). Since C_E^{ML} is independent of θ, the two functionals have the same minimizers. □

The exact martingale loss is the natural theoretical offline objective. In the implementation, however, the European offline baseline is trained with a closely related payoff-regression loss. For each fixed time t, define the discounted terminal target by

$$Y_t := e^{-r(T-t)}g_K(S_T). \tag{B.4}$$

This random variable is the terminal payoff discounted back to the current time t.

By the risk-neutral valuation formula,

$$V_E(t,S_t) = \mathbb{E}^{\mathbb{Q}}\left[Y_t \mid \mathcal{F}_t\right], \tag{B.5}$$

so the European value at time t is the conditional expectation of the discounted terminal payoff.

The implemented regression loss is

$$\mathcal{L}_E^{\mathrm{reg}}(\theta) := \frac{1}{2}\,\mathbb{E}^{\mathbb{Q}}\left[\int_0^T \left|Y_t - V_{E,\theta}(t,S_t)\right|^2 dt\right]. \tag{B.6}$$

This is the objective minimized in the offline European baseline step.

The exact martingale loss and the implemented regression loss are closely related, but they are not identical when $r > 0$. Using (B.4), we obtain

$$\mathcal{L}_E^{\mathrm{ML}}(\theta) = \frac{1}{2}\,\mathbb{E}^{\mathbb{Q}}\left[\int_0^T e^{-2rt}\left|Y_t - V_{E,\theta}(t,S_t)\right|^2 dt\right]. \tag{B.7}$$

Therefore, the two losses differ only by the deterministic time weight e^{-2rt}.

To analyze the implemented loss, define the mean-square value error by

$$\mathrm{MSVE}_E(\theta) := \mathbb{E}^{\mathbb{Q}}\left[\int_0^T \left|V_E(t,S_t) - V_{E,\theta}(t,S_t)\right|^2 dt\right]. \tag{B.8}$$

This is the time-integrated value error corresponding to the implemented regression objective.

Theorem B.2 (Regression loss and mean-square value error)**.** *The implemented regression loss* $\mathcal{L}_E^{\mathrm{reg}}(\theta)$ *and the mean-square value error* $\mathrm{MSVE}_E(\theta)$ *have the same minimizers. More*

precisely,

$$\mathcal{L}_E^{\mathrm{reg}}(\theta) = C_E^{\mathrm{reg}} + \frac{1}{2}\,\mathrm{MSVE}_E(\theta), \tag{B.9}$$

where

$$C_E^{\mathrm{reg}} = \frac{1}{2}\,\mathbb{E}^{\mathbb{Q}}\left[\int_0^T |Y_t - V_E(t,S_t)|^2\,dt\right],$$

is independent of θ, so it does not affect the minimization problem.

Proof. Fix $t \in [0,T]$. By (B.5), $V_E(t,S_t)$ is the conditional expectation of Y_t given $\mathcal{F}_t$. Hence it is the L^2-projection of Y_t onto the space of square-integrable $\mathcal{F}_t$-measurable random variables.

Since $V_E(t,S_t) = \mathbb{E}^{\mathbb{Q}}[Y_t \mid \mathcal{F}_t]$ is the $L^2(\mathbb{Q})$-orthogonal projection of Y_t onto the space of square-integrable $\mathcal{F}_t$-measurable random variables (Kallenberg, 2002), and since $V_{E,\theta}(t,S_t)$ is $\mathcal{F}_t$-measurable, the Pythagorean projection identity gives

$$\mathbb{E}^{\mathbb{Q}}\left[|Y_t - V_{E,\theta}(t,S_t)|^2\right] = \mathbb{E}^{\mathbb{Q}}\left[|Y_t - V_E(t,S_t)|^2\right] + \mathbb{E}^{\mathbb{Q}}\left[|V_E(t,S_t) - V_{E,\theta}(t,S_t)|^2\right].$$

Integrating this identity over $t \in [0,T]$ and multiplying by $1/2$ gives (B.9). Since C_E^{reg} is independent of θ, the two functionals have the same minimizers. □

For computation, let $0 = t_0 < t_1 < \cdots < t_N = T$ be a time grid, with $\Delta t_n = t_{n+1} - t_n$. The implemented regression loss is approximated by

$$\widehat{\mathcal{L}}_E^{\mathrm{reg}}(\theta) = \frac{1}{2}\,\mathbb{E}^{\mathbb{Q}}\left[\sum_{n=0}^{N-1} \left|e^{-r(T-t_n)} g_K(S_T) - V_{E,\theta}(t_n,S_{t_n})\right|^2 \Delta t_n\right]. \tag{B.10}$$

This is the time-discretized European offline objective used in the Black–Scholes implementation.

In the European fixed-maturity problem, the absence of running rewards makes the discounted terminal payoff a natural observable target. Accordingly, although the exact martingale loss of Jia and Zhou (2022) remains theoretically valid, the implementation uses a regression loss to this target. This directly trains the value function $V_{E,\theta}$ and is justified by the conditional-expectation identity (B.5).

B.2 Bernoulli Policy-Improvement Rule

This appendix section derives the Bernoulli policy-improvement rule used in the American premium-learning component. For fixed (t, s) and a fixed value argument w, the local Hamiltonian is

$$\mathcal{H}_{\widehat{E}}(t, s, \pi; w) = -\Gamma\pi w + \Gamma\pi\widehat{g}(t, s) - \lambda H(\pi), \qquad \pi \in (0, 1),$$

where

$$H(\pi) = \pi \log \pi + (1 - \pi) \log(1 - \pi).$$

The maximization is one-dimensional, since (t, s) and w are held fixed.

Differentiating the entropy term with respect to π gives

$$\frac{d}{d\pi}H(\pi) = \log\left(\frac{\pi}{1-\pi}\right).$$

Therefore,

$$\frac{d}{d\pi}\mathcal{H}_{\widehat{E}}(t,s,\pi;w) = -\Gamma w + \Gamma\widehat{g}(t,s) - \lambda\log\left(\frac{\pi}{1-\pi}\right).$$

The second derivative is

$$\frac{d^2}{d\pi^2}\mathcal{H}_{\widehat{E}}(t,s,\pi;w) = -\lambda\left(\frac{1}{\pi} + \frac{1}{1-\pi}\right) < 0, \qquad \pi \in (0,1).$$

Hence the Hamiltonian is strictly concave in π, and any solution of the first-order condition is the unique maximizer.

The first-order condition is

$$-\Gamma w + \Gamma\widehat{g}(t,s) - \lambda\log\left(\frac{\pi}{1-\pi}\right) = 0.$$

Equivalently,

$$\log\left(\frac{\pi}{1-\pi}\right) = \lambda^{-1}\Gamma(\widehat{g}(t,s) - w).$$

Exponentiating both sides gives

$$\frac{\pi}{1-\pi} = \exp\left(\lambda^{-1}\Gamma(\widehat{g}(t,s) - w)\right).$$

Solving for π yields

$$\pi^*_{\widehat{E}}(t,s;w) = \frac{1}{1+\exp(\lambda^{-1}\Gamma(w-\widehat{g}(t,s)))}. \tag{B.11}$$

This is the policy-improvement rule stated in (3.15).

B.3 Offline American Martingale Loss

Recall that for a fixed policy π, the premium-learning problem becomes a policy-evaluation problem. The fixed-policy premium value is

$$J^\pi_{\widehat{E}}(t,s) = \mathbb{E}^{\mathbb{Q}}\left[\int_t^T e^{-r(v-t)} R^\pi_{t,v}\rho^\pi(v,S_v)\,dv \;\middle|\; S_t = s\right].$$

Equivalently, $J^\pi_{\widehat{E}}$ satisfies the fixed-policy equation

$$\begin{cases} (\mathcal{L}J^\pi_{\widehat{E}})(v,s) - rJ^\pi_{\widehat{E}}(v,s) - \Gamma\pi_v J^\pi_{\widehat{E}}(v,s) + \rho^\pi(v,s) = 0, & (v,s)\in[t,T)\times(0,\infty), \\ J^\pi_{\widehat{E}}(T,s) = 0, & s>0. \end{cases} \tag{B.12}$$

Here $\rho^\pi(v,s) = \Gamma\widehat{g}(v,s)\pi(v,s) - \lambda H(\pi(v,s))$, with the contract dependence suppressed for readability.

Define the reward-adjusted process

$$M^\pi_{t,v} := e^{-r(v-t)}R^\pi_{t,v}J^\pi_{\widehat{E}}(v,S_v) + \int_t^v e^{-r(\ell-t)}R^\pi_{t,\ell}\rho^\pi(\ell,S_\ell)\,d\ell, \qquad v\in[t,T]. \tag{B.13}$$

We now show that $M^{\pi}_{t,v}$ is a martingale in the future time variable v. Applying Itô's formula to $J^{\pi}_{\widehat{E}}(v, S_v)$, using

$$\frac{d}{dv} R^{\pi}_{t,v} = -\Gamma \pi_v R^{\pi}_{t,v},$$

and applying the product rule to

$$e^{-r(v-t)} R^{\pi}_{t,v} J^{\pi}_{\widehat{E}}(v, S_v),$$

the drift term becomes

$$e^{-r(v-t)} R^{\pi}_{t,v} \left[(\mathcal{L} J^{\pi}_{\widehat{E}})(v, S_v) - r J^{\pi}_{\widehat{E}}(v, S_v) - \Gamma \pi_v J^{\pi}_{\widehat{E}}(v, S_v) \right] dv.$$

By the fixed-policy equation (B.12), this drift is equal to

$$-e^{-r(v-t)} R^{\pi}_{t,v} \rho^{\pi}(v, S_v)\, dv.$$

The accumulated-running-term part of (B.13) contributes

$$e^{-r(v-t)} R^{\pi}_{t,v} \rho^{\pi}(v, S_v)\, dv,$$

so the drift terms cancel. Therefore $M^{\pi}_{t,v}$ is a local martingale, and under the integrability conditions assumed in the policy-evaluation framework it is a martingale.

Since the fixed-policy premium has zero terminal value, $J^{\pi}_{\widehat{E}}(T, S_T) = 0$, the terminal

martingale value is

$$M^{\pi}_{t,T} = \int_t^T e^{-r(\ell-t)} R^{\pi}_{t,\ell} \rho^{\pi}(\ell, S_\ell)\, d\ell. \tag{B.14}$$

Thus, in the premium formulation, the terminal martingale value is the accumulated discounted running term generated by the fixed policy.

Replacing $J^{\pi}_{\widehat{E}}$ in (B.13) by the neural approximation U_{A,θ_A} gives the approximate reward-adjusted process

$$M^{\pi,\theta_A}_{t,v} := e^{-r(v-t)} R^{\pi}_{t,v} U_{A,\theta_A}(v, S_v) + \int_t^v e^{-r(\ell-t)} R^{\pi}_{t,\ell} \rho^{\pi}(\ell, S_\ell)\, d\ell. \tag{B.15}$$

The offline martingale loss compares this approximate process with the terminal martingale value:

$$\mathcal{L}^{\text{off}}_{A,\pi}(\theta_A) := \frac{1}{2}\mathbb{E}^{\mathbb{Q}}\left[\int_t^T \left| M^{\pi}_{t,T} - M^{\pi,\theta_A}_{t,v} \right|^2 dv \;\middle|\; S_t = s\right]. \tag{B.16}$$

This loss has the same projection structure as the martingale loss used for the European baseline. Since $M^{\pi}_{t,v}$ is a martingale,

$$M^{\pi}_{t,v} = \mathbb{E}^{\mathbb{Q}}\left[M^{\pi}_{t,T} \mid \mathcal{F}_v\right].$$

Therefore, by the L^2-projection identity for conditional expectation,

$$\mathcal{L}^{\text{off}}_{A,\pi}(\theta_A) = C^{\text{off}}_{A,\pi} + \frac{1}{2}\,\text{WMSVE}_{A,\pi}(\theta_A), \tag{B.17}$$

where

$$\mathrm{WMSVE}_{A,\pi}(\theta_A) := \mathbb{E}^{\mathbb{Q}}\left[\int_t^T \left|e^{-r(v-t)}R^{\pi}_{t,v}(J^{\pi}_{\widehat{E}}(v,S_v) - U_{A,\theta_A}(v,S_v))\right|^2 dv \;\middle|\; S_t = s\right], \quad \text{(B.18)}$$

and

$$C^{\mathrm{off}}_{A,\pi} := \frac{1}{2}\mathbb{E}^{\mathbb{Q}}\left[\int_t^T \left|M^{\pi}_{t,T} - M^{\pi}_{t,v}\right|^2 dv \;\middle|\; S_t = s\right]$$

is independent of θ_A. Hence, for a fixed policy π, minimizing the offline martingale loss is equivalent to minimizing the weighted mean-square error between the fixed-policy premium value $J^{\pi}_{\widehat{E}}$ and the neural premium approximation U_{A,θ_A}.

We now derive the discrete residual used in the implementation. Let $t = t_0 < t_1 < \cdots < t_L = T$ be the exercise grid, with $t_\ell = t + \ell\Delta t$. On this grid, the weighting process is computed forward by

$$R^{\pi}_{t,t_{j+1}} = R^{\pi}_{t,t_j}\left(1 - \Gamma\pi_{t_j}\Delta t\right), \qquad R^{\pi}_{t,t} = 1. \quad \text{(B.19)}$$

The values R^{π}_{t,t_j} are then used in the backward residual.

Using (B.14), the terminal martingale value is

$$M^{\pi}_{t,T} = \int_t^T e^{-r(v-t)}R^{\pi}_{t,v}\rho^{\pi}(v,S_v)\,dv.$$

From (B.15), at grid time t_ℓ,

$$M_{t,t_\ell}^{\pi,\theta_A} = e^{-r(t_\ell - t)} R_{t,t_\ell}^{\pi} U_{A,\theta_A}(t_\ell, S_{t_\ell}) + \int_t^{t_\ell} e^{-r(v-t)} R_{t,v}^{\pi} \rho^{\pi}(v, S_v)\, dv.$$

Subtracting these two expressions gives

$$M_{t,T}^{\pi} - M_{t,t_\ell}^{\pi,\theta_A} = \int_{t_\ell}^{T} e^{-r(v-t)} R_{t,v}^{\pi} \rho^{\pi}(v, S_v)\, dv - e^{-r(t_\ell - t)} R_{t,t_\ell}^{\pi} U_{A,\theta_A}(t_\ell, S_{t_\ell}).$$

Thus the accumulated running term before t_ℓ cancels, leaving only the future accumulated running term from t_ℓ to maturity and the discounted network prediction at t_ℓ.

Approximating the remaining integral by a left-point Riemann sum gives the discrete offline residual

$$G_\ell^{\pi}(\theta_A) := \sum_{j=\ell}^{L-1} e^{-r(t_j - t)} R_{t,t_j}^{\pi} \rho^{\pi}(t_j, S_{t_j})\, \Delta t - e^{-r(t_\ell - t)} R_{t,t_\ell}^{\pi} U_{A,\theta_A}(t_\ell, S_{t_\ell}). \tag{B.20}$$

This is the time-discretized version of $M_{t,T}^{\pi} - M_{t,t_\ell}^{\pi,\theta_A}$.

Finally, the time integral in the continuous loss is approximated by the grid sum,

$$\int_t^T \left| M_{t,T}^{\pi} - M_{t,v}^{\pi,\theta_A} \right|^2 dv \approx \sum_{\ell=0}^{L-1} |G_\ell^{\pi}(\theta_A)|^2\, \Delta t.$$

Therefore, the implemented Black–Scholes offline American option premium-learning loss

is

$$\widehat{\mathcal{L}}^{\text{off}}_{A,\pi}(\theta_A) := \frac{1}{2}\mathbb{E}^{\mathbb{Q}}\left[\sum_{\ell=0}^{L-1}(G^{\pi}_{\ell}(\theta_A))^2\,\Delta t \;\middle|\; S_t = s\right]. \tag{B.21}$$

In practice, this expectation is replaced by the empirical average over simulated risk-neutral paths.

B.4 Squared TD Residual and Quadratic Variation

This appendix section explains why the online method is based on martingale orthogonality rather than on directly minimizing a squared one-step TD residual.

For the European baseline, the one-step residual of the approximation is

$$\delta^{\theta}_{E,n} = e^{-r\Delta t_n}V_{E,\theta}(t_{n+1}, S_{t_{n+1}}) - V_{E,\theta}(t_n, S_{t_n}).$$

A naive mean-square TD objective would minimize a quantity of the form

$$\mathbb{E}^{\mathbb{Q}}\left[\sum_{n=0}^{N-1}\left(\delta^{\theta}_{E,n}\right)^2\right].$$

The difficulty is that, even when the approximation is exact, the one-step residual is not zero pathwise. To see this, let $V_{E,\theta} = V_E$, and define

$$\delta^{*}_{E,n} := e^{-r\Delta t_n}V_E(t_{n+1}, S_{t_{n+1}}) - V_E(t_n, S_{t_n}).$$

Since the discounted European value process $M_t^E = e^{-rt}V_E(t, S_t)$ is a martingale, the true one-step residual satisfies

$$\mathbb{E}^{\mathbb{Q}}\left[\delta^*_{E,n} \mid \mathcal{F}_{t_n}\right] = 0.$$

However, the martingale property does not imply that the squared residual is zero. In general,

$$\mathbb{E}^{\mathbb{Q}}\left[(\delta^*_{E,n})^2 \mid \mathcal{F}_{t_n}\right] \neq 0,$$

because the true discounted value process still has random martingale fluctuations. This can be seen directly from Itô's formula. For the true European value,

$$dM_t^E = e^{-rt}\sigma S_t \frac{\partial V_E}{\partial s}(t, S_t)\, dW_t.$$

Therefore, over a small time interval,

$$M^E_{t+\Delta t} - M^E_t \approx e^{-rt}\sigma S_t \frac{\partial V_E}{\partial s}(t, S_t)\Delta W_t.$$

Squaring and taking conditional expectation gives

$$\mathbb{E}^{\mathbb{Q}}\left[\left(M^E_{t+\Delta t} - M^E_t\right)^2 \mid \mathcal{F}_t\right] \approx e^{-2rt}\sigma^2 S_t^2 \left(\frac{\partial V_E}{\partial s}(t, S_t)\right)^2 \Delta t.$$

Thus the squared one-step residual measures the local martingale variance. In the continuous-

time limit, this quantity is connected to the quadratic variation

$$d\langle M^E\rangle_t = e^{-2rt}\sigma^2 S_t^2 \left(\frac{\partial V_E}{\partial s}(t, S_t)\right)^2 dt.$$

Consequently, minimizing the squared one-step residual would penalize unavoidable martingale fluctuations, rather than directly minimizing $|V_E(t, S_t) - V_{E,\theta}(t, S_t)|^2$. The online method therefore uses the martingale condition in its conditional-mean form. Instead of forcing the residual itself to be small in square, it imposes moment conditions of the form

$$\mathbb{E}^{\mathbb{Q}}\left[\nabla_\theta V_{E,\theta}(t_n, S_{t_n})\, \delta^\theta_{E,n}\right] = 0,$$

which is the martingale-orthogonality condition used in the online update.

B.5 Semi-Gradient Form of the Online CTD(0) Update

This appendix section explains why the update in (3.24) is called a semi-gradient update. The point is that the CTD(0) update is not obtained by ordinary gradient descent on the squared one-step residual. If one minimized $\frac{1}{2}(\delta^\theta_{E,n})^2$ directly, then the full gradient would be

$$\nabla_\theta \left[\frac{1}{2}(\delta^\theta_{E,n})^2\right] = \delta^\theta_{E,n} \nabla_\theta \delta^\theta_{E,n}.$$

Since

$$\nabla_\theta \delta^\theta_{E,n} = e^{-r\Delta t_n} \nabla_\theta V_{E,\theta}(t_{n+1}, S_{t_{n+1}}) - \nabla_\theta V_{E,\theta}(t_n, S_{t_n}),$$

this full-gradient direction differentiates through both the next-step value and the current value.

The CTD(0) update used in this thesis follows a different principle. It comes from the martingale-orthogonality moment condition

$$\mathbb{E}^{\mathbb{Q}} \left[\nabla_\theta V_{E,\theta}(t_n, S_{t_n}) \delta^\theta_{E,n} \right] = 0.$$

Thus the update direction is

$$\nabla_\theta V_{E,\theta}(t_n, S_{t_n}) \delta^\theta_{E,n}.$$

Only the current value approximation is differentiated. The residual $\delta^\theta_{E,n}$ is treated as a realized scalar during the update.

This distinction matters because the squared one-step residual is not a direct value-function error objective in stochastic continuous time. Even for the true value function, the martingale increment has zero conditional mean but generally nonzero conditional variance. Therefore, minimizing the squared residual would penalize unavoidable martingale fluctuations. In the continuous-time limit, this is connected to the quadratic variation of the martingale rather than to the mean-square value error (Jia and Zhou, 2022).

B.6 Online American One-Step Residual

This appendix section derives the one-step residual used in the online American premium-learning update. The notation follows Section 3.2. The starting time is fixed at t, and the exercise grid is $t = t_0 < t_1 < \cdots < t_L = T$, where $t_{\ell+1} - t_\ell = \Delta t$.

For a fixed Bernoulli policy π, the reward-adjusted process associated with the neural premium approximation is

$$M_{t,v}^{\pi,\theta_A} = e^{-r(v-t)} R_{t,v}^{\pi} U_{A,\theta_A}(v, S_v) + \int_t^v e^{-r(u-t)} R_{t,u}^{\pi} \rho^{\pi}(u, S_u)\, du, \qquad v \in [t, T].$$

The online method uses the one-step martingale condition over $[t_\ell, t_{\ell+1}]$. Therefore, we examine the increment

$$M_{t,t_{\ell+1}}^{\pi,\theta_A} - M_{t,t_\ell}^{\pi,\theta_A}.$$

By the definition of $M_{t,v}^{\pi,\theta_A}$,

$$\begin{aligned} M_{t,t_{\ell+1}}^{\pi,\theta_A} - M_{t,t_\ell}^{\pi,\theta_A} =& e^{-r(t_{\ell+1}-t)} R_{t,t_{\ell+1}}^{\pi} U_{A,\theta_A}(t_{\ell+1}, S_{t_{\ell+1}}) \\ &- e^{-r(t_\ell-t)} R_{t,t_\ell}^{\pi} U_{A,\theta_A}(t_\ell, S_{t_\ell}) \\ &+ \int_{t_\ell}^{t_{\ell+1}} e^{-r(u-t)} R_{t,u}^{\pi} \rho^{\pi}(u, S_u)\, du. \end{aligned}$$

For compactness, write

$$U_\ell := U_{A,\theta_A}(t_\ell, S_{t_\ell}), \qquad U_{\ell+1} := U_{A,\theta_A}(t_{\ell+1}, S_{t_{\ell+1}}), \qquad \rho_\ell^{\pi} := \rho^{\pi}(t_\ell, S_{t_\ell}),$$

Approximating the integral over $[t_\ell, t_{\ell+1}]$ by the left endpoint gives

$$\int_{t_\ell}^{t_{\ell+1}} e^{-r(u-t)} R^\pi_{t,u} \rho^\pi(u, S_u)\, du \approx e^{-r(t_\ell - t)} R^\pi_{t,t_\ell} \rho^\pi_\ell \, \Delta t.$$

Thus

$$M^{\pi,\theta_A}_{t,t_{\ell+1}} - M^{\pi,\theta_A}_{t,t_\ell} \approx e^{-r(t_\ell - t)} R^\pi_{t,t_\ell} \left[e^{-r\Delta t} \frac{R^\pi_{t,t_{\ell+1}}}{R^\pi_{t,t_\ell}} U_{\ell+1} - U_\ell + \rho^\pi_\ell \, \Delta t \right].$$

The common prefactor $e^{-r(t_\ell - t)} R^\pi_{t,t_\ell}$ is positive and $\mathcal{F}_{t_\ell}$-measurable. Removing it does not change the zero conditional-mean martingale condition. Therefore, the normalized one-step increment is based on the term inside the brackets.

The next-step premium is multiplied by the financial discount factor $e^{-r\Delta t}$. Using the first-order expansion

$$e^{-r\Delta t} = 1 - r\Delta t + O((\Delta t)^2),$$

we obtain

$$e^{-r\Delta t} \frac{R^\pi_{t,t_{\ell+1}}}{R^\pi_{t,t_\ell}} U_{\ell+1} = \frac{R^\pi_{t,t_{\ell+1}}}{R^\pi_{t,t_\ell}} U_{\ell+1} - r\Delta t \frac{R^\pi_{t,t_{\ell+1}}}{R^\pi_{t,t_\ell}} U_{\ell+1} + O((\Delta t)^2).$$

To first order, the financial discount correction may be evaluated at the left endpoint, which gives

$$-r\Delta t \frac{R^\pi_{t,t_{\ell+1}}}{R^\pi_{t,t_\ell}} U_{\ell+1} \approx -r U_\ell \Delta t.$$

In the Black–Scholes implementation, the penalty-induced survival factor is discretized

using the forward-Euler recurrence. Therefore,

$$\frac{R^{\pi}_{t,t_{\ell+1}}}{R^{\pi}_{t,t_\ell}} = 1 - \Gamma \pi_{t_\ell} \Delta t.$$

Combining the first-order financial-discount approximation with this survival-factor recurrence gives the implemented online American residual

$$\delta^{\theta_A}_{\ell} := \frac{R^{\pi}_{t,t_{\ell+1}}}{R^{\pi}_{t,t_\ell}} U_{\ell+1} - U_\ell + (\rho^{\pi}_{\ell} - rU_\ell)\Delta t. \tag{B.22}$$

Equivalently, using

$$\rho^{\pi}_{\ell} = \Gamma \widehat{g}(t_\ell, S_{t_\ell})\pi_{t_\ell} - \lambda H(\pi_{t_\ell}),$$

this residual can be written as

$$\begin{aligned}\delta^{\theta_A}_{\ell} =& \frac{R^{\pi}_{t,t_{\ell+1}}}{R^{\pi}_{t,t_\ell}} U_{A,\theta_A}(t_{\ell+1}, S_{t_{\ell+1}}) - U_{A,\theta_A}(t_\ell, S_{t_\ell}) \\ &+ \Big(\Gamma \widehat{g}(t_\ell, S_{t_\ell})\pi_{t_\ell} - \lambda H(\pi_{t_\ell}) - rU_{A,\theta_A}(t_\ell, S_{t_\ell})\Big)\Delta t.\end{aligned}$$

This is the residual used in the Black–Scholes online American option premium-learning update. The Merton-specific residual is given in Section 3.3.

References

Bates, David S. (1996). "Jumps and Stochastic Volatility: Exchange Rate Processes Implicit in Deutsche Mark Options". In: The Review of Financial Studies 9.1, pp. 69–107. DOI: 10.1093/rfs/9.1.69.

Becker, Sebastian, Patrick Cheridito, and Arnulf Jentzen (2019). "Deep Optimal Stopping". In: Journal of Machine Learning Research 20.74, pp. 1–25.

Becker, Sebastian, Patrick Cheridito, Arnulf Jentzen, and Timo Welti (2021). "Solving High-Dimensional Optimal Stopping Problems Using Deep Learning". In: European Journal of Applied Mathematics 32.3, pp. 470–514. DOI: 10.1017/S0956792521000073.

Ben-Ameur, Hatem, Michèle Breton, and Pierre L'Ecuyer (2002). "A Dynamic Programming Procedure for Pricing American-Style Asian Options". In: Management Science 48.5, pp. 625–643. DOI: 10.1287/mnsc.48.5.625.7803.

Black, Fischer and Myron Scholes (1973). "The pricing of options and corporate liabilities". In: J. Polit. Econ. 81.3, pp. 637–654. ISSN: 0022-3808,1537-534X. DOI: 10.1086/260062. URL: https://doi-org.proxy.bib.uottawa.ca/10.1086/260062.

Bloch, Daniel Alexandre (Aug. 2023). American Options: Models and Algorithms. https://ssrn.com/abstract=4532952. DOI: 10.2139/ssrn.4532952.

Boyle, Phelim P. (1977). "Options: A Monte Carlo Approach". In: Journal of Financial Economics 4.3, pp. 323–338. DOI: 10.1016/0304-405X(77)90005-8.

Cai, Ning and Steven G. Kou (2011). "Option Pricing Under a Mixed-Exponential Jump Diffusion Model". In: Management Science 57.11, pp. 2067–2081. DOI: 10.1287/mnsc.1110.1393.

Carr, Peter et al. (2002). "The Fine Structure of Asset Returns: An Empirical Investigation". In: The Journal of Business 75.2, pp. 305–332. DOI: 10.1086/338705.

Carriere, Jacques F. (1996). "Valuation of the Early-Exercise Price for Options Using Simulations and Nonparametric Regression". In: Insurance: Mathematics and Economics 19.1, pp. 19–30. DOI: 10.1016/S0167-6687(96)00004-2.

Cox, John C., Stephen A. Ross, and Mark Rubinstein (1979). "Option Pricing: A Simplified Approach". In: Journal of Financial Economics 7.3, pp. 229–263. DOI: 10.1016/0304-405X(79)90015-1.

Dai, Min et al. (2025). "Learning to Optimally Stop Diffusion Processes, with Financial Applications". In: arXiv: 2408.09242 [math.OC]. URL: https://arxiv.org/abs/2408.09242.

Duan, Jin-Chuan (1995). "The GARCH Option Pricing Model". In: Mathematical Finance 5.1, pp. 13–32. DOI: 10.1111/j.1467-9965.1995.tb00099.x.

Forsyth, Peter A. and Kenneth R. Vetzal (2002). "Quadratic Convergence for Valuing American Options Using a Penalty Method". In: SIAM Journal on Scientific Computing 23.6, pp. 2095–2122. DOI: 10.1137/S1064827500382324.

Friedman, Avner (1982). Variational principles and free-boundary problems. A Wiley-Interscience Publication. Pure and Applied Mathematics. John Wiley & Sons, Inc., New York, pp. ix+710. ISBN: 0-471-86849-3.

Gao, Xuefeng, Lingfei Li, and Xun Yu Zhou (Mar. 2026). "Reinforcement Learning for Jump-Diffusions, With Financial Applications". In: Mathematical Finance. Advance online publication. DOI: 10.1111/mafi.70027. URL: https://doi.org/10.1111/mafi.70027.

Glasserman, Paul (2004). Monte Carlo methods in financial engineering. Vol. 53. Applications of Mathematics (New York). Stochastic Modelling and Applied Probability. Springer-Verlag, New York, pp. xiv+596. ISBN: 0-387-00451-3.

Han, Jiequn, Arnulf Jentzen, and Weinan E (2018). "Solving high-dimensional partial differential equations using deep learning". In: Proc. Natl. Acad. Sci. USA 115.34, pp. 8505–8510. ISSN: 0027-8424,1091-6490.

DOI: 10.1073/pnas.1718942115. URL: https://doi-org.proxy.bib.uottawa.ca/10.1073/pnas.1718942115.

Heston, Steven L. (1993). "A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options". In: The Review of Financial Studies 6.2, pp. 327–343. DOI: 10.1093/rfs/6.2.327.

Heston, Steven L. and Saikat Nandi (2000). "A Closed-Form GARCH Option Valuation Model". In: The Review of Financial Studies 13.3, pp. 585–625. ISSN: 0893-9454. DOI: 10.1093/rfs/13.3.585. URL: https://doi.org/10.1093/rfs/13.3.585.

Huré, Côme et al. (2021). "Deep Neural Networks Algorithms for Stochastic Control Problems on Finite Horizon: Convergence Analysis". In: SIAM Journal on Numerical Analysis 59.1, pp. 525–557. DOI: 10.1137/20M1316640.

Jia, Yanwei and Xun Yu Zhou (2022). "Policy evaluation and temporal-difference learning in continuous time and space: a martingale approach". In: J. Mach. Learn. Res. 23, Paper No. [154], 55. ISSN: 1532-4435,1533-7928.

Kallenberg, Olav (2002). Foundations of Modern Probability. Second. Probability and Its Applications. New York: Springer-Verlag. ISBN: 978-0-387-95313-7. DOI: 10.1007/978-1-4757-4015-8.

Karatzas, Ioannis and Steven E. Shreve (1991). Brownian motion and stochastic calculus. Second. Vol. 113. Graduate Texts in Mathematics. Springer-Verlag, New York, pp. xxiv+470. ISBN: 0-387-97655-8. DOI: 10.1007/978-1-4612-0949-2. URL: https://doi-org.proxy.bib.uottawa.ca/10.1007/978-1-4612-0949-2.

Kloeden, Peter E. and Eckhard Platen (1992). Numerical solution of stochastic differential equations. Vol. 23. Applications of Mathematics (New York). Springer-Verlag, Berlin, pp. xxxvi+632. ISBN: 3-540-54062-8. DOI: 10.1007/978-3-662-12616-5. URL: https://doi-org.proxy.bib.uottawa.ca/10.1007/978-3-662-12616-5.

Kor, Peyman, Reidar B. Bratvold, and Aojie Hong (2024). "Is Reinforcement Learning Good at American Option Valuation?" In: Algorithms 17.9. ISSN: 1999-4893. DOI: 10.3390/a17090400. URL: https://www.mdpi.com/1999-4893/17/9/400.

Kou, Steven G. (2002). "A Jump-Diffusion Model for Option Pricing". In: Management Science 48.8, pp. 1086–1101. DOI: 10.1287/mnsc.48.8.1086.166.

Lagoudakis, Michail G. and Ronald Parr (2004). "Least-squares policy iteration". In: J. Mach. Learn. Res. 4.6, pp. 1107–1149. ISSN: 1532-4435,1533-7928. DOI: 10.1162/1532443041827907. URL: https://doi-org.proxy.bib.uottawa.ca/10.1162/1532443041827907.

Li, Nan (2022). "An Iteration Algorithm for American Options Pricing Based on Reinforcement Learning". In: Symmetry 14.7. ISSN: 2073-8994. DOI: 10.3390/sym14071324. URL: https://www.mdpi.com/2073-8994/14/7/1324.

Li, Yuxi, Csaba Szepesvári, and Dale Schuurmans (2009). "Learning exercise policies for American options". In: International Conference on Artificial Intelligence and Statistics. PMLR, pp. 352–359.

Longstaff, Francis A. and Eduardo S. Schwartz (2001). "Valuing American Options by Simulation: A Simple Least-Squares Approach". In: The Review of Financial Studies 14.1, pp. 113–147. DOI: 10.1093/rfs/14.1.113.

McDonald, Robert L. and Mark Schroder (1998). "A parity result for American options". In: Journal of Computational Finance 1, pp. 5–13. URL: https://api.semanticscholar.org/CorpusID:151042314.

Merton, Robert C. (1973). "Theory of rational option pricing". In: Bell J. Econom. and Management Sci. 4, pp. 141–183. ISSN: 0005-8556,2325-5323.

— (1976). "Option Pricing When Underlying Stock Returns Are Discontinuous". In: Journal of Financial Economics 3.1–2, pp. 125–144. DOI: 10.1016/0304-405X(76)90022-2.

Øksendal, Bernt (2003). Stochastic differential equations. Sixth. Universitext. An introduction with applications. Springer-Verlag, Berlin, pp. xxiv+360. ISBN: 3-540-04758-1. DOI: 10.1007/978-3-642-14394-6. URL: https://doi-org.proxy.bib.uottawa.ca/10.1007/978-3-642-14394-6.

Peskir, Goran and Albert Shiryaev (2006). Optimal stopping and free-boundary problems. Lectures in Mathematics ETH Zürich. Birkhäuser Verlag, Basel, pp. xxii+500. ISBN: 978-3-7643-2419-3; 3-7643-2419-8.

Shannon, C. E. (1948). "A mathematical theory of communication". In: Bell System Tech. J. 27, pp. 379–423, 623–656. ISSN: 0005-8580. DOI: 10.1002/j.1538-7305.1948.tb01338.x. URL: https://doi-org.proxy.bib.uottawa.ca/10.1002/j.1538-7305.1948.tb01338.x.

Shreve, Steven E. (2004). Stochastic calculus for finance. II. Springer Finance. Continuous-time models. Springer-Verlag, New York, pp. xx+550. ISBN: 0-387-40101-6.

Sutton, Richard S. (1988). "Learning to predict by the methods of temporal differences". In: Machine Learning 3.1, pp. 9–44. ISSN: 0885-6125. DOI: 10.1007/bf00115009.

Sutton, Richard S. and Andrew G. Barto (2018). Reinforcement learning: an introduction. Second. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, pp. xxii+526. ISBN: 978-0-262-03924-6.

Teto, Manal (2025). "Efficient Calibration and Greeks Estimation for a Panel of American Options Using a Stochastic Dynamic Program". MA thesis. Ottawa, Canada: University of Ottawa. URL: https://hdl.handle.net/10393/50319.

Tsitsiklis, John N. and Benjamin Van Roy (2001). "Regression Methods for Pricing Complex American-Style Options". In: IEEE Transactions on Neural Networks 12.4, pp. 694–703. DOI: 10.1109/72.935083.