High-dimensional American Option Pricing using Neural Networks

Kylle Lansangan Joel Leo Xueyin Yang
Jianyang Bi

July 2025

Contents

Abstract

This study attempts to develop a supervised deep learning framework for estimation of the fair value V_0 of American options in high-dimensional settings. The Weak Variance Alpha-Gamma (WVAG) process is employed to model log returns, leveraging the flexibility of the Variance Gamma process and the ability of the WVAG process to capture more complex dependence structures among underlying asset prices. Parameter estimation is carried out via Digital Moment Estimation (DME) on historical market data. The resulting parameter estimates are fed into a deep residual neural network (ResNet) that maps WVAG parameters to the fair price V_0. The network trained via Monte Carlo simulation yields accurate price estimates while being significantly more time-efficient comapred to numerical methods such as the Longstaff-Schwartz algorithm. This demostrates the potential of combining deep learning with weakly subordinated Lévy models to enable efficient pricing in high-dimensional financial markets.

1 Introduction

Some financial derivatives are able to be exercised before its time of maturity. One type of option that allows for this is the American option. As a result, the fair price of the American option V_0 is a solution to an optimal stopping problem, unlike European options. The formulation of the optimal stopping problem will be made known in the next section.

It has been proposed that Artificial Neural Networks (ANN) can be used to overcome the curse of dimensionality that makes high-dimensional optimal stopping problems difficult to solve (Becker et al., 2018, 2019; Kohler et al., 2010; Ye and Zhang, 2019). Such machine learning methods are alternatives to numerical methods such as the Longstaff-Schwartz (LS) and Tsitsiklis-Roy (TR) algorithms proposed by Longstaff and Schwartz, 2001; Tsitsiklis and Roy, 1999, which are Monte Carlo Least-Squares Regression algorithms that estimate the optimal value V_0. We will be using the LS algorithm as a benchmark to compare the results of our ANN. According to Ruf and Wang, 2019, different papers approach the optimal stopping problems and

as a result, the use of neural networks varies. The details of our Machine Learning approach are further elaborated on in the proceeding sections.

In the paper mentioned above, there is a limitation: that is, the assumed dynamics of the underlying assets. For example, in Kohler et al., 2010, the Black-Scholes model is used to model the underlying asset prices. Such models may not be able to fully capture the empirical features present in financial market observations. As a result, we will explore the use of more expressive models. A good starting point would be the pure-jump Variance Gamma (VG) process proposed in Madan et al., 1998. This model has become popular in finanical modelling due to many reasons, such as the incorporation of jump behaviour, greater control over skewness and kurtosis, and heavier tails (Fischer et al., 2023). This added expressiveness will be able to allow greater capture of the features of real-world financial data.

To extend this to the multivariate case to study such processes at higher dimensions, we will consider weakly subordinated multivariate Lévy processes, in particular, Weak Variance Alpha-Gamma (WVAG) processes. The WVAG process is a type of multivariate VG process subordinated by the alpha-gamma process mentioned in Luciano and Semeraro, 2010b, allowing for the further modelling of (codependent) volatility. An important thing to note is that the fact that the WVAG process involves weak subordination (Buchmann et al., 2016), which allows the multivariate Brownian motion to have a non-diagonal covariance matrix (Buchmann et al., 2018). This allows for richer codependence structure between the underlying assets.

Using the WVAG process to model underlysing asset log-prices and supervised learning, this study will attempt to demonstrate that the neural networks used in american option pricing can prove to be effective on real world data when complemented with a model of greater complexity and expressiveness.

2 Mathematical Framework

2.1 WVAG Model

We will make the assumption that asset log-prices follow a WVAG process. We first begin our discussion with some defintions from Buchmann et al., 2018.

Definition 2.1 (Gamma Subordinator) *Let $a, b > 0$. A univariate subordinator $G \sim \Gamma_S(a, b)$ is called a gamma subordinator if its marginal $G(t)$, $t \geq 0$, is gamma distributed with shape parameter at and rate parameter b. If $a = b$, we call G a standard gamma subordinator and write $G \sim \Gamma_S(b) := \Gamma^S(b, b)$.*

Definition 2.2 (Alpha-Gamma Subordinator) *Assume $n \geq 2$. Let $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n_+$ and let $G_0, \ldots, G_n$ be independent gamma subordinators such that*

$$G_0 \sim \Gamma_S(a, 1), \quad G_k \sim \Gamma_S(\beta_k, 1/\alpha_k), \quad 1 \leq k \leq n,$$

where $a > 0$, $a\alpha_k < 1$, and $\beta_k := \frac{1-a\alpha_k}{\alpha_k}$.

An alpha-gamma subordinator $T \sim AG^n_S(a, \alpha)$ with parameters α, a is then defined by

$$T \overset{d}{=} G_0\alpha + (G_1, \ldots, G_n),$$

Then T has correlated components with marginals $T_k \sim \Gamma_S(1/\alpha_k)$, $1 \leq k \leq n$.

Definition 2.3 (Weak Variance-Alpha-Gamma process) *Assume $n \geq 2$. Let $\mu \in \mathbb{R}^n$ and $\Sigma \in \mathbb{R}^{n\times n}$ be a covariance matrix (not necessarily diagonal). The process $X \sim WVAG_n(a, \alpha, \mu, \Sigma)$ is called a weak variance-alpha-gamma process with parameters (a, α, μ, Σ) if*

$$X \overset{d}{=} BM_n(\mu, \Sigma) \odot AG^n_S(a, \alpha),$$

where $BM_n(\mu, \Sigma)$ denotes n-dimensional Brownian motion with drift μ and covariance Σ, and $\odot$ denotes the weak subordination operation.

However, observe that there is no deterministic drift. Although this is a common theoretical assumption, this assumption may not hold when we attempt to fit the WVAG process to real world data. Thus, we will add deterministic drift in the following definition:

Definition 2.4 (Weak Variance-Alpha-Gamma (deterministic drift)) *Let $Y_t \sim WVAG_n(a, \alpha, \theta, \Sigma)$ for some $a, \alpha, \theta, \Sigma$, and let $\mu \in \mathbb{R}^n$ and $G_t \sim AG^n_S(a, \alpha)$*

Then our WVAG process with deterministic drift $X_t = \mu t + Y_t$ can be expressed as follows:

$$X_t = \mu t + \theta * G_t + \Sigma^{1/2} W_{G_t},$$

Where $*$ denotes the Hadamard product, and $\Sigma = \Sigma^{1/2}(\Sigma^{1/2})^t$. Note that this means we are assuming that Σ is positive definite.

The price at time t, S_t is then:

$$S_t = S_0 \frac{\exp(X_t)}{\mathbb{E}[\exp(X_t)]}$$

Under the risk-neutral measure.

For more information on the WVAG process and weak subordination, refer to Buchmann and Lu, 2020; Buchmann et al., 2016, 2018; Luciano and Semeraro, 2010b.

We use a representation similar to (2.18) found in Fischer et al., 2023, which will be more useful in simulating the WVAG process, which in general consists of:

1. Simulating n standard brownian motions each subordinated by one component of the alpha-gamma process separately. This gives us the subordinated standard brownian motion vector W_{G_t} at each time t. the value of G_t is stored for Step 3.

2. Applying a linear transformation to the subordinated brownian motion vector at each time t by pre-multiplying by $\Sigma^{1/2}$

3. Calculating the jump drift and the deterministic drift, $\theta * G_t$ and μt respectively.

2.1.1 Economic interpretation of the WVAG process

The alpha-gamma process allows for two componenents (Luciano and Semeraro, 2010b):

1. An idosyncractic subordinator component, which express intra-market shocks which affect one asset.

2. A common subordinator component, which express inter-market shocks that affect multiple assets.

The weak subordination proposed allows that Σ need not be diagonal (Buchmann et al., 2018), this allows the capturing of more complicated correlation and covariance structure. From Definition 2.4, we may intepret each

asset being influenced by some linear combination of noise sources represented by the brownian motions.

These features of the WVAG process add to its ability to capture multivariable asset price processes.

2.2 Optimal Stopping problem

The following equation describes the optimal stopping problem. V_0 is the fair value of the American option and ψ is some discounted payoff function.

$$V_0 = \sup_{\tau \in \mathcal{T}[0,T]} E(\psi(S_\tau)) \tag{1}$$

Where $\mathcal{T}$ is the class of all [0, T]-valued stopping times.

The goal of our ANN is estimate a function $f : \Omega \to \mathbb{R}$, where f returns the V_0 of an American option with underlying assets governed by some WVAG process specified by its parameters $\theta \in \Theta$, and some specified payoff function ψ. Similarly to Kohler et al., 2010, one strength of this method is that the payoff function can be chosen to be more complicated, but for the purposes of our testing, we will choose a simple payoff function.

3 Model Calibration

The first part of our solution to the optimal stopping problem uses the Digital Moment Estimation (DME) method from Buchmann et al., 2018 to search for parameters that allow a WVAG process to fit real-world data (assuming there is a good fit). However, the aforementioned paper only performs this estimation in 2 dimensions, and we attempt to perform DME on 5-dimensional data.

3.1 Data preprocessing

We scraped interday data of the following stocks and indices from Yahoo Finance, and performed basic data cleaning by excludidng all rows with invalid entries (i.e. we exclude the asset price vector at points in time where there is at least 1 invalid entry):

1. S&P 500

2. Tesla, Inc.
3. Exxon Mobil Corporation
4. JPMorgan Chase & Co.
5. Walmart Inc.

3.2 Parameter Estimation

We follow the Digital Moment Estimation Method mentiomned in Buchmann et al., 2018. For brevity, we omit the specifications of the Parameter Estimation.

3.3 Regularisation

However, we would like to note that there are many problems with achieving accurate parameter estimation in high dimensions (since there are 5 asset prices, dim = 5).

One main drawback of our method was that heavy regularisation was required to fit the data to any reasonable accuracy. This regularisation largely included penalties from momenta deviation. As a result, parameter estimation shifted towards more of momenta fitting.

3.4 Results

The result of the digital moment estimation for the data were as follows:

$$a = 0.4999999528562542$$

$$\alpha = \begin{pmatrix} 0.36975349 \\ 0.20347236 \\ 0.18888967 \\ 0.21789399 \\ 0.37051509 \end{pmatrix}$$

$$\theta = \begin{pmatrix} -0.00134792 \\ -0.00100919 \\ -0.0006225 \\ -0.00029067 \\ -0.00125155 \end{pmatrix}$$

$$\mu = \begin{pmatrix} 0.00067216 \\ 0.0004758 \\ 0.00058607 \\ 0.00078873 \\ 0.00068173 \end{pmatrix}$$

$$\sigma = \begin{pmatrix} 0.01714273 \\ 0.01392764 \\ 0.02233846 \\ 0.01484495 \\ 0.02280182 \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 2.93873281 & 1.62738071 & 1.51874343 & 9.11748808 & 1.02292216 \\ 1.62738071 & 1.93979088 & 1.97338314 & 7.50920193 & 1.72601260 \\ 1.51874343 & 1.97338314 & 4.99006869 & 3.73675917 & 1.92699469 \\ 9.11748808 & 7.50920193 & 3.73675917 & 2.20372549 & 4.08775443 \\ 1.02292216 & 1.72601260 & 1.92699469 & 4.08775443 & 5.19922845 \end{pmatrix} \times 10^{-4}$$

Figure 1 shows the scatter plots showing randomly generated points from the estimated parameters orange and the actual datapoints in blue.

4 Deep Neural Network

4.1 Introduction and Motivation

We study the problem of pricing an American basket-put option under a sophisticated multivariate stochastic model (the WVAG process) calibrated to real-world data. Exact valuation via the *sup-of-*$\mathbb{E}$ Monte Carlo estimator is accurate but computationally expensive: it requires simulating high-dimensional stochastic paths, normalising to enforce moment constraints, computing discounted pathwise payoffs, and taking a supremum over exercise times. This makes it prohibitive for real-time deployment, large-scale scenario analysis, or embedding inside nested optimisation or risk pipelines. Our motivation is therefore twofold. First, we seek to drastically reduce per-query latency by replacing the heavy simulation with a neural surrogate that approximates the Monte Carlo estimate in milliseconds instead of hundreds or tens of thousands of milliseconds, enabling scalability and responsiveness. Second, we acknowledge that a standalone surrogate inevitably exhibits systematic bias across the input space; to mitigate this, we decompose the problem in log-price space and learn the residual between the true Monte Carlo

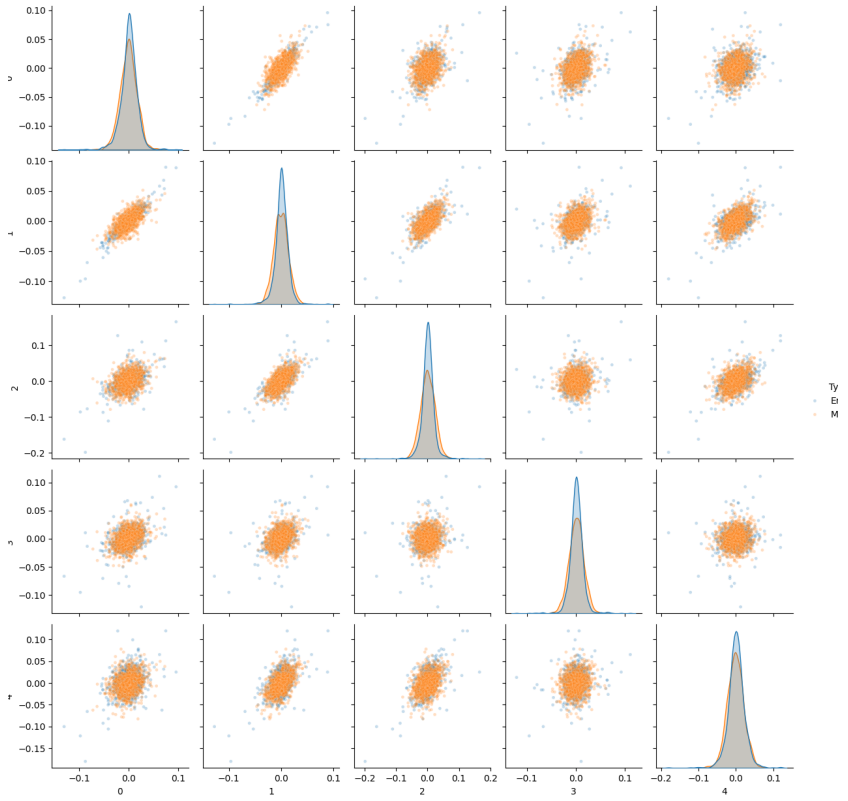

Figure 1: Pairwise scatter plots showing the correlation structure of the calibrated WVAG process parameters

log-price and the base surrogate's prediction. This multiplicative correction captures structured errors while retaining the surrogate's speed. To anchor our approach in reality, we fit the underlying WVAG parameters to empirical data and perform a sanity check using the classical Least-Squares Monte Carlo (LSMC) method with those estimated parameters, providing a lower-bound consistency check on the American option price. Together, these elements form a hybrid framework that balances accuracy and efficiency: a fast surrogate, a learned log-domain correction for bias, and validation against principled Monte Carlo baselines.

4.2 Training Data Generation

The training data were generated in a controlled, supervised fashion to span a wide but realistic region of the parameter space, with the underlying ranges derived from the real-world parameter estimates and then modestly widened to provide robustness and avoid overfitting to narrow empirical pockets. To efficiently cover this high-dimensional space, Latin hypercube sampling was employed: spot and strike levels, drift μ, loading θ, covariance entries (via the full matrix), subordinator parameters α and a, interest rate r, and time-step $\mathrm{d}t$ were sampled jointly in a stratified way so that each marginal was well represented without requiring prohibitively many draws. The sampled raw parameters were unpacked per instance, with the full covariance reshaped and stabilised via a safe Cholesky factorisation (adding jitter if needed and projecting to positive definiteness as a fallback) to ensure numerical robustness. For each sample, the *sup-of-*$\mathbb{E}$ Monte Carlo price estimate was computed using the high-fidelity stochastic simulator, producing the ground truth label; the surrogate input vector was constructed from the primitives (including the Cholesky factor) and its corresponding log-price formed the target after taking the logarithm (with a small offset for numerical safety). Feature normalisation statistics (means and standard deviations) were computed across the generated dataset, and inputs were standardised accordingly. A fixed random seed ensured reproducibility throughout generation, and the resulting dataset—consisting of tens of thousands of examples with their augmented, normalised inputs and log-price targets—was prepared into batched loaders with pinning and multiple workers for efficient downstream training.

4.3 Feature Engineering

A critical component in achieving strong surrogate performance was the extensive feature engineering, which was carried out to expose relevant invariances, disentangle nonlinear dependencies, and provide the networks with more directly usable signals. The raw input comprises primitives such as the spot vector S_0, drift μ, loading parameters θ, the Cholesky factor of the covariance (capturing correlated noise structure), subordinator parameters α and a, and scalars including strike K, interest rate r, and time-step $\mathrm{d}t$. From these, derived features were constructed deliberately to encode known structural effects: normalised spots and log-spots introduce scale invariance and stabilise magnitude differences across samples; moneyness and log-moneyness

(and their squares) directly reflect nonlinear payoff sensitivity to relative strike versus scale; cross-interactions of moneyness/log-moneyness with μ, θ, and α allow the model to capture how the option value shifts depending on both the level and "tilt" of the underlying dynamics; and scaled versions of moneyness and log-moneyness tied back to spot scale help the network reconcile absolute and relative effects. This expansion markedly increased input dimensionality but was purposeful: by precomputing these interactions and invariances, both the base surrogate and the residual correction stage are relieved of having to discover them from raw data alone, speeding convergence and improving sample efficiency. Furthermore, clamping inputs at inference to the empirical 1st and 99th percentile quantiles of the augmented training distribution ensures the surrogate operates within familiar regimes, mitigating the risk of erratic extrapolation when encountering out-of-distribution inputs.

4.4 Surrogate Model Design & Training Methodology

The surrogate model is constructed as a two-stage corrective cascade to balance speed with accuracy.

Stage 1: Base Surrogate in Log-Space. The first stage is a feedforward residual network (ResNet-style) tasked with predicting the *log* of the *sup-of-*$\mathbb{E}$ price estimate. Working in log-space is deliberate: the log-price surface is empirically smoother, has reduced relative variance, and compresses dynamic range, all of which ease optimisation compared to raw price space. The base network begins with an embedding layer that projects the high-dimensional, augmented input into a hidden representation via a linear transformation, batch normalisation, a ReLU nonlinearity, and dropout. This front end stabilises feature scale, introduces nonlinearity, and regularises before deeper processing. The core of the model is a sequence of residual blocks; each block contains two linear layers with intervening batch normalisation, ReLU activations, and dropout, plus a skip connection (identity or learned projection). This structure enables modelling complex nonlinear dependencies while preserving gradient flow and mitigating degradation in deeper stacks. A final linear layer outputs a scalar log-price prediction. Training uses a Smooth L1 loss (Huber-style with $\beta = 0.1$). Optimisation is performed with Adam with weight decay (10^{-5}) under a OneCycleLR schedule (peak LR 3×10^{-3}, pct_start = 0.1, div_factor = 25, final_div_factor = 10^4) and then decays it

with aggressive initial/final scaling to encourage convergence to flatter minima. To further improve generalisation, stochastic weight averaging (SWA) is employed after 75% of the epochs, followed by BN statistics recomputation. Mixed precision, gradient scaling, and gradient clipping (max norm 1.0) are used.

Stage 2: Heteroscedastic Residual Correction. Recognising systematic bias relative to the high-fidelity estimator, a second-stage corrective model is introduced: a heteroscedastic residual network. It takes as input the normalised surrogate features concatenated with two summary statistics, normalised scale and normalised moneyness, and predicts the standardised log-residual, i.e., the difference between the true Monte Carlo log-price and the surrogate's log prediction (normalised to zero mean and unit variance). "Heteroscedastic" refers to the model's ability to represent input-dependent uncertainty: instead of assuming constant residual noise, it outputs both a predicted mean correction $\mu(x)$ and a log-variance $\log \sigma^2(x)$, defining a Gaussian distribution for the residual. This allows the model to be confident (low variance) where the surrogate is already accurate and to express higher uncertainty in harder regions of the input space. Architecturally, the residual network consists of a shared trunk of linear layers with ReLU activations and dropout (hidden widths [512,256,128], dropout=0.01), followed by two separate heads producing the normalised residual mean and log-variance. The log-variance output is clamped to a safe range to avoid numerical instability. The primary loss is the heteroscedastic negative log-likelihood:

$$\mathcal{L}_{\text{NLL}} \;=\; \tfrac{1}{2}\left[\frac{(r_{\text{true}} - \mu(x))^2}{\sigma^2(x)} \;+\; \log \sigma^2(x)\right],$$

which jointly encourages accurate corrections while scaling uncertainty appropriately. Two auxiliary terms augment this likelihood:

1. A relative mean absolute error (**rel-MAE**) from the Monte Carlo ground truth is computed in price space after applying the correction and exponentiating. This penalises cases where log-space corrections yield disproportionately large multiplicative errors in price space, enforcing fidelity where it matters for downstream decisions.

2. A smoothed L1 (Huber) penalty on the normalised residual mean itself, serving as an ℓ_1-style regulariser with tunable sensitivity to outliers (controlled by the Huber delta β)

The total objective is:

$$\mathcal{L} \;=\; \mathcal{L}_{\mathrm{NLL}} \;+\; \lambda_{\mathrm{mae}} \cdot \text{rel-MAE} \;+\; \lambda_{\ell_1} \cdot \mathrm{Huber}(r_{\mathrm{norm}}, \beta).$$

Training uses AdamW with OneCycleLR (peak LR 3×10^{-3}, pct_start = 0.1, div_factor = 10, final_div_factor = 50) for up to 100 epochs; gradients are clipped (max norm 3.0); early stopping is based on validation NLL; deterministic seeding ensures reproducibility.

4.5 Hyperparameter Grid Search

To balance bias, variance and calibration in the residual correction, a grid search was conducted over the regularisation weights and smoothing parameters: the ℓ_1 coefficient λ_{ℓ_1}, the relative-MAE weight λ_{mae}, and the Huber delta β. Models were reinitialised per trial and trained with early stopping based on validation NLL. The best configuration: $\lambda_{\ell_1} = 0.5$, $\lambda_{\mathrm{mae}} = 1$, $\beta = 0.2$ yielded a relative mean absolute error of 2.0386% on the full training dataset.

4.6 Baseline Sanity Check using LSMC

As a sanity check and to ground the surrogate's estimates, the classical Least-Squares Monte Carlo (LSMC, Longstaff-Schwartz) method was applied using the same WVAG simulator, with parameters calibrated from real-world data. A polynomial basis (constant, first and second order in the basket average) was used to regress continuation values at in-the-money time steps, performing backward induction to approximate the optimal early exercise policy. The output of LSMC constitutes a lower bound on the true American option value. With real-world calibrated parameters, LSMC produced a price of approximately 4534.96, slightly below the *sup-of-*$\mathbb{E}$ Monte Carlo estimate of 4544.53, as expected. This ordering is consistent with theory (*sup-of-*$\mathbb{E}$ providing an upper/more optimistic estimate via taking the supremum over exercise times in expectation, while LSMC approximates a suboptimal policy), and the tight gap validates that both stochastic estimators are coherent for the empirical regime under study.

4.7 Evaluation

Quantitatively, the surrogate plus residual correction pipeline achieves strong aggregate performance: relative mean absolute error = 2.0386% and rela-

tive mean squared error = 0.0590% against the high-fidelity *sup-of-*$\mathbb{E}$ estimates across the sampled dataset, with average inference time of approximately 3.255 milliseconds - orders of magnitude faster than the raw *sup-of-*$\mathbb{E}$ Monte Carlo (about 436.989 ms in the bulk dataset and 927.427 ms in the real-world single-parameter run), while the surrogate+residual pipeline took about 3.483 ms.

The LSMC lower-bound estimator produced a price very close to the *sup-of-*$\mathbb{E}$ reference (≈ 4534.96 vs. 4544.53), but at dramatically higher computational cost (≈ 40.18 seconds, reflecting full backward induction over 2^{15} steps, with the backward pass itself completing 32,767 iterations). For the real-world calibrated parameters, the base surrogate predicts $V_{\text{nn}} \approx 4186.76$, and after residual correction $V_{\text{corr}} \approx 4132.70$; by contrast, the *sup-of-*$\mathbb{E}$ estimate is ≈ 4544.53. Thus, on this particular parameter setting the corrected surrogate is about 9.0621% lower than the high-fidelity reference, revealing a systematic gap despite strong aggregate metrics and low latency.

5 Discussion

Several limitations merit discussion. The 2% rel-MAE over the synthetic training/evaluation set reflects strong average fidelity, but the 9.06% relative error between the corrected neural prediction and the *sup-of-*$\mathbb{E}$ on the real-world calibrated parameters reveals sensitivity to distributional shift or localised bias. Potential causes include: training over a broad parameter space that underrepresents the specific empirical regime; limited capacity/regularisation of the residual model; non-Gaussian structure in log-residuals not fully captured by the heteroscedastic Gaussian assumption; auxiliary relative-MAE penalty trades off bias and variance in a way that can leave persistent systematic under-correction; and possible noise or small upward bias in the *sup-of-*$\mathbb{E}$ ground truth itself, complicating exact alignment.

Possible remedies include localised fine-tuning or importance-weighted sampling around empirical regimes; ensembling surrogate/residual pairs; physics-informed constraints (e.g., monotonicity in strike, no-arbitrage bounds); mixture-density residuals; active learning to sample high-residual scenarios; and uncertainty-calibrated hybrid evaluation that triggers full Monte Carlo when uncertainty is high.

6 Conclusion

We presented a two-stage neural surrogate framework for approximating a computationally expensive *sup-of-*$\mathbb{E}$ estimator for American basket-put pricing under a WVAG process. Through feature engineering, heteroscedastic residual modelling, and systematic hyperparameter search, the method achieves low average errors ($\sim 2\%$ rel-MAE) while reducing inference time by orders of magnitude. LSMC provided a sanity check on real-world calibrated parameters, corroborating consistency between stochastic methods. The observed gap in the real-world setting highlights areas for refinement, including localised fine-tuning and uncertainty-aware hybrid evaluation. Overall, the approach demonstrates a practical path to deploying learned surrogates in derivative pricing pipelines where speed and reasonable accuracy must coexist.

References

Becker, S., Cheridito, P., & Jentzen, A. (2018). Deep optimal stopping. *J. Mach. Learn. Res.*, *20*, 74:1–74:25.

Becker, S., Cheridito, P., Jentzen, A., & Welti, T. (2019). Solving high-dimensional optimal stopping problems using deep learning. *European Journal of Applied Mathematics*, *32*, 470–514.

Buchmann, B., & Lu, K. W. (2020). Necessity of weak subordination for some strongly subordinated lévy processes. *Journal of Applied Probability*, *58*, 868–879.

Buchmann, B., Lu, K. W., & Madan, D. (2016). Weak subordination of multivariate lévy processes and variance generalised gamma convolutions. *Bernoulli*.

Buchmann, B., Lu, K. W., & Madan, D. (2018). Calibration for weak variance-alpha-gamma processes. *Methodology and Computing in Applied Probability*, *21*, 1151–1164.

Fischer, A., Gaunt, R. E., & Sarantsev, A. (2023). The variance-gamma distribution: A review. *Statistical Science*.

Kohler, M., Krzyżak, A., & Todorović, N. (2010). Pricing of high-dimensional american options by neural networks. *Mathematical Finance*, *20*.

Longstaff, F., & Schwartz, E. S. (2001). Valuing american options by simulation: A simple least-squares approach. *The Finance*.

Luciano, E., & Semeraro, P. (2010b). Multivariate time changes for lévy asset models: Characterization and calibration. *J. Comput. Appl. Math.*, *233*, 1937–1953.

Madan, D., Carr, P., & Chang, E. C. (1998). The variance gamma process and option pricing. *Review of Finance*, *2*, 79–105.

Ruf, J., & Wang, W. (2019). Neural networks for option pricing and hedging: A literature review. *PSN: Other Political Methods: Quantitative Methods (Topic)*.

Tsitsiklis, J., & Roy, B. V. (1999). Optimal stopping of markov processes: Hilbert space theory, approximation algorithms, and an application to pricing high-dimensional financial derivatives. *IEEE Trans. Autom. Control.*, *44*, 1840–1851.

Ye, T., & Zhang, L. (2019). Derivatives pricing via machine learning. *Boston: Information Technology (Topic)*.