Machine Learning & Numerical Methods for Convertible Bond Valuation

HL Team 2 :
Chengjia Dong
Srikari Rallabandi
Kewei Xiang
Michael Wattelet

Master of Science in Financial Engineering
University of Illinois at Urbana-Champaign

Sponsored by Houlihan Lokey

Spring 2025

Abstract

This report explores advanced methods for valuing convertible bonds by examining the classic model: the Tsiveriotis-Fernandes (TF) model, while addressing key considerations such as credit risk, stock volatility, and market liquidity. Beginning with a thorough data cleaning process, this work presents the implementation and error analysis of the TF model, followed by a discussion of opportunities to improve model accuracy through enhanced parameter estimation and refined numerical schemes. Machine learning (ML) and reinforcement learning (RL) techniques are introduced to replicate or refine the PDE solutions of the TF model, demonstrating how data-driven methods can reduce computational overhead and capture complex market behaviors. The findings reveal that ML-based models can approach PDE-level accuracy, though challenges remain in the calibration and handling of boundary artifacts. Future directions for refining PDE boundary conditions include integrating advanced reinforcement learning methods and incorporating real-world factors, such as liquidity and stochastic credit spreads. This work aims to develop a more robust and scalable framework for convertible bond valuation.

Keywords: Convertible Bond, Tsiveriotis-Fernandes Model, Machine Learning, Reinforcement Learning, Regression, Implied Volatility, Replication.

Acknowledgments

The authors express great gratitude to their corporate sponsor, Andrew MacNamara, Qi (Kelsey) Guo, Jialing (Peter) Zhu from Houlihan Lokey, for their generous support and valuable insights on the work which help the authors' growth throughout the semester. Their contributions, data access, and technical discussions have been instrumental in advancing this work in convertible bond valuation as well as the academic learning of the authors. Thank you also to Chong Zhao and Ruichen Zhao from Houlihan Lokey, who also offered a lot of insightful help.

In addition, the authors express their gratitude to the University of Illinois Urbana-Champaign Master of Science in Financial Engineering program for providing the opportunity to work with Houlihan Lokey and comprehensive support throughout the semester.

Executive Summary

The goal of this project was to improve convertible-bond valuation, focusing on the Tsiveriotis–Fernandes (TF) framework and data-driven enhancements. Convertible bonds' hybrid nature demands models that incorporate equity option features alongside credit risk; yet classical TF implementations remain sensitive to inputs and computationally intensive.

A finite-difference Crank–Nicolson solver was built for the TF partial differential equations, with stability controlled by CFL-type ratios and convergence verified through a relative-change metric. Diagnostic testing uncovered a systematic proportional bias and nonlinear errors tied to implied volatility and credit-spread inputs. A post-pricing linear correction cut root-mean-squared error between the TF model and market price by about 90 percent, and decision-tree analysis identified interest rates, implied volatility, time-to-maturity, and credit spreads as dominant drivers for a dynamic volatility adjustment.

To accelerate valuation, supervised learning models were trained on a synthetic dataset of four million scenarios drawn from empirically calibrated joint distributions. A feed-forward neural network reproduced TF prices with roughly 0.7 percent mean absolute percentage error and placed more than three-quarters of predictions within a 1 percent band, while delivering micro-second inference times.

Reinforcement-learning experiments (tabular Q-learning, least-squares policy iteration, policy-gradient, and actor–critic variants) were framed as optimal stopping problems but primarily focused on replicating the pricing behavior of the TF model rather than learning explicit exercise strategies. While agents reduced pricing error after extended training, their performance remained sensitive to reward design and the realism of market simulations.

Continuing work will refine TF boundary conditions, integrate liquidity-adjusted volatility estimates, explore physics-informed neural networks for PDE surrogacy, improve Machine Learning model accuracy, and validate the hybrid numerical–ML framework against historical bond data to ensure robustness in production-level applications.

Contents

1 Introduction

1.1 Houlihan Lokey

This project is sponsored by Houlihan Lokey, a global investment bank specializing in mergers and acquisitions (M&A), capital markets, financial restructuring, and financial and valuation advisory services. The authors are working with their Financial and Valuation Advisory Business Line, which specializes in the valuation for complex derivatives, including convertible bonds. Private equity firms, hedge funds, investment managers, and financial institutions rely on Houlihan Lokey for their independent third-party valuation services.

1.2 Introduction to Convertible Bonds

Convertible bonds are hybrid financial instruments that combine features of both debt and equity. They are issued as traditional bonds, paying periodic interest and returning principal at maturity, but include an embedded option that allows the bondholder to convert the bond into a predetermined number of shares of the issuing company's stock. This convertible feature provides potential upside participation in the company's equity performance while offering downside protection through fixed-income payments. As a result, convertible bonds can be attractive to investors seeking a balance between risk and return. From an issuer's perspective, convertible bonds can be cost-effective, as they allow for lower coupon rates in exchange for the equity conversion option. However, their valuation is more complex than that of standard bonds, because it must consider interest rate risk, equity price dynamics, and credit risk, as well as the embedded optionalities such as calls, puts, and conversion rights.

1.3 Existing Models for Convertible Bond Valuation

The Tsiveriotis-Fernandes model separates the value of a convertible bond into a bond (cash) component, which is subject to credit risk, and an equity component, which is assumed to be default-free. This decomposition allows the model to apply a credit spread only to the risky part of the instrument. Let V be the total value of the convertible bond, B the bond component, and E the equity component, so that $V = B + E$.

The TF model is governed by a system of coupled partial differential equations (PDEs). The PDE for the total convertible bond value $u(S,t)$ is shown in equation 1.

$$CB : \frac{\partial u}{\partial t} + \tfrac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2} + (r-d)\, S \frac{\partial u}{\partial S} - r\, u + r\, v + f(t) = 0, \tag{1}$$

The PDE for the cash-only part $v(S,t)$ is shown in equation 2.

$$COCB : \frac{\partial v}{\partial t} + \tfrac{1}{2}\sigma^2 S^2 \frac{\partial^2 v}{\partial S^2} + (r-d)\, S \frac{\partial v}{\partial S} - (r + r_c)\, v + f(t) = 0. \tag{2}$$

The parameters are defined as follows: S is the underlying stock price, σ is the annualized volatility of the stock's returns, r is the continuously compounded risk-free interest rate, d is the continuous dividend yield on the stock, r_c is the credit spread applied to the cash-only (bond) component, $u(S,t)$ is the total CB value combining both cash and equity components, $v(S,t)$ is the cash-only component (COCB) representing

default-risky payments, and $f(t)$ is the source term for discrete cash flows (coupons), implemented via delta functions.

This approach is realistic because it distinguishes the credit-risky bond portion from the default-free equity portion, but the numerical implementation is relatively demanding and sensitive to boundary conditions.

To solve these PDEs using an explicit finite difference method, the continuous derivatives are replaced by finite difference approximations. For example, if the asset price S and time t are discretized using steps ΔS and Δt with

$$S_i = i\,\Delta S \quad , \quad t_n = n\,\Delta t,$$

then the explicit finite difference scheme for the bond component $B(S,t)$ can be represented using equation 3.

$$v_i^{n+1} = v_i^n + \Delta t\left[\tfrac{1}{2}\,\sigma^2 S_i^2 \frac{v_{i+1}^n - 2v_i^n + v_{i-1}^n}{\Delta S^2} + (r-d)\,S_i \frac{v_{i+1}^n - v_{i-1}^n}{2\,\Delta S} - (r+r_c)\,v_i^n\right], \quad (3)$$

$$u_i^{n+1} = u_i^n + \Delta t\left[\tfrac{1}{2}\,\sigma^2 S_i^2 \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta S^2} + (r-d)\,S_i \frac{u_{i+1}^n - u_{i-1}^n}{2\,\Delta S} - r\,u_i^n + r\,v_i^n\right]. \quad (4)$$

In these equations, the derivatives are approximated at time level n (i.e., the known current values), allowing the solution to be advanced step-by-step. Note that the stability of this explicit method requires that Δt be sufficiently small relative to ΔS. In practice, the below condition must be satisfied to prevent divergence or oscillations.

$$\Delta t \le \frac{\Delta S^2}{2\sigma^2}$$

Conversion price c and conversion ratio a enter through the free-boundary (early-conversion) conditions.

$$a = \frac{1}{c}$$

The payoff at maturity $t = T$ is shown in the following expressions.

$$u(S,T) = \max\big(a\,S,\ B\big), \quad v(S,T) = \begin{cases} B, & S < c \\ 0, & S \ge c \end{cases}$$

At any intermediate time t, the solution must also satisfy the following expressions.

$$u(S,t)\ \ge\ a\,S, \quad v(S,t) = 0 \quad \text{whenever } S \ge c$$

Thus "moneyness" S/c appears in the boundary conditions: if the stock price exceeds the conversion price, immediate exercise forces u down to the conversion value aS and kills off the cash-only component v.

1.4 Project Motivation

Despite its rigorous foundation, the Tsiveriotis–Fernandes model exhibits systematic biases and practical limitations when applied to market data. In particular, raw implied volatilities tend to produce consistently overvalued convertible bond prices—a

phenomenon observed across a sample of 85 bonds—and require a bespoke calibration step to correct. Moreover, the finite-difference solution of the TF PDEs is computationally expensive: achieving the stability and accuracy demanded by small spatial (ΔS) and temporal (Δt) discretizations can render each valuation prohibitively slow. These challenges motivate our development of a machine-learning surrogate that both incorporates a volatility calibration procedure to eliminate pricing bias and delivers TF-level accuracy at a fraction of the computational cost.

1.5 Project Scope and Objectives

The goal of this project is to enhance the accuracy and efficiency of convertible bond valuation, focusing primarily on improving the Tsiveriotis-Fernandes model. This work builds upon preliminary works (section 2) completed by previous MSFE practicum project groups at the University of Illinois Urbana-Champaign. The Tsiveriotis-Fernandes model is implemented and evaluated on convertible bond data (sections 3 and 4) with improvements discussed in section 5 (Proportional Bias and Implied Volatility parameter estimation). Section 6 discusses synthetic data generation, and section 7 shows how supervised learning can be used to replicate the TF model to reduce computational overhead without compromising accuracy. Section 8 discusses replication using reinforcement learning. By integrating these data-driven methods, the objective to develop a more robust and scalable framework that addresses real-world market conditions and improved precision in pricing convertible bonds is completed. Future work (section 10) is also presented to expand on the work completed by the authors.

2 Preliminary Works

The senior team's preliminary work focused on developing and testing numerical methods for convertible bond valuation using three models: the Tsiveriotis-Fernandes (TF) model, the Goldman Sachs (GS) model, and the Hull reduced-form model. They implemented these models while incorporating key factors, such as, credit risk, stock volatility, and default probability. They then evaluated the model accuracies by comparing the outputted model price to market price estimates using real data from Bloomberg. Their analysis included sensitivity tests on stock price, volatility, and credit spread, highlighting areas where model performance diverged from observed bond prices.

To improve accuracy, they explored calibration techniques and gradient descent adjustments, leading to significant reductions in pricing errors. The TF model demonstrated generalizability across different bond types, while the GS and Hull models benefited from parameter tuning. Their final refinements reduced mean squared error (MSE) by nearly 40%, showcasing the effectiveness of numerical enhancements in convertible bond valuation.

3 Data Manipulation & Analysis

The dataset that used for convertible bond pricing contains eighty-five convertible bonds, totaling to 47577 rows of time series data. The data is organized into multiple sheets, with sheet 1 listing the names of the bonds and their corresponding dividend rates, sheet 2 listing the interpolated treasury rates for different periods, and the remaining

sheets listing properties and features of each of the 85 convertible bonds, such as coupon rate, stock price, and other relevant metrics.

The original dataset had several issues that required cleaning. Some features, such as bond volume, were missing or left blank. Some features had missing data points caused misalignment in the dates. Others had extreme outlier values, such as sudden spikes in bond volume which were obvious typos in the dataset.

To address these issues, the data was cleaned and organized into separate files. This step was crucial for ensuring compatibility with machine learning algorithms and improving computational efficiency. Missing data values were removed, and the dates were aligned to ensure consistency across all features. Extreme outliers were also removed to prevent skewed results.

4 Tsiveriotis-Fernandes Model Evaluation

4.1 TF Model Implementation

The TF model extends the Black-Scholes framework to value convertible bonds with embedded credit risk. It decomposes the total bond value, $u(S,t)$, into a "clean" component, $v(S,t)$, (free of credit risk) and also a credit adjustment term, $(u-v)$. This separation allows explicit modeling of the credit spread r_c while avoiding direct estimation in the original PDE.

The classical Black-Scholes equation, shown in equation 5, struggles with the unknown credit spread. The non-linear features, such as early exercise, are captured in the $f(u,S,t)$ term.

$$\frac{\partial u}{\partial t} + \frac{S^2\sigma^2}{2}\frac{\partial^2 u}{\partial S^2} + rS\frac{\partial u}{\partial S} - (r+r_c)u + f(u,S,t) = 0 \tag{5}$$

To combat this problem, the TF model uses two coupled PDEs, which resolve the credit spread ambiguity. The PDE referring to the value of the convertible bond (CB) is shown in equation 6 and the PDE referring to the value of the conversion option embedded in the convertible bond (COCB) is shown in equation 7.

$$\text{CB: } \frac{\partial u}{\partial t} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 u}{\partial S^2} + rS\frac{\partial u}{\partial S} - r(u-v) - (r+r_c)v + f(t) = 0, \tag{6}$$

$$\text{COCB: } \frac{\partial v}{\partial t} + \frac{1}{2}\sigma^2 S^2\frac{\partial^2 v}{\partial S^2} + rS\frac{\partial v}{\partial S} - (r+r_c)v + f(t) = 0. \tag{7}$$

In equation 6, u represents the value of the convertible bond, v denotes the value of the embedded conversion option, S denotes the price of the underlying stock, σ represents the volatility of the underlying stock, r represents the risk-free interest rate, and r_c is the cost of carry associated with the convertible bond. Additionally, $f(t)$ is an external input, which may account for factors such as dividends or other cash flows that affect the bond. In equation 7, v represents the value of the conversion option, with all other parameters being the same as equation 6.

The key features of the TF model include the decoupling of u and v, in which u and v are solved separately for numerical stability, the embedded credit spread, in which r_c is incorporated explicitly without the need for ad-hoc estimation, and the simplified calibration with the linearized source term $f(t)$ to improve convergence.

The TF (Tsiveriotis-Fernandes) model was implemented as outlined by previous work. The implementation involved defining a function to solve coupled partial differential equations (PDEs) using the Finite Difference Method (FDM), utilizing iterative loops within the FDM solver, which were optimized using the `numba` library's Just-In-Time (JIT) compilation for faster execution and adding progress tracking using the `tqdm` package to monitor the runtime and estimate completion time.

4.2 Error Analysis

Complete code of the TF model was provided by the senior team. The TF model takes as input the principal, time to maturity, risk-free rate, dividend, CDS, coupon rate, coupon frequency, S steps, t steps, call price, put price, stock price and outputs the model price. To evaluate the distribution of error of TF model, an error analysis was performed on a sample bond (TEVA 0 $\frac{1}{4}$ 02/01/26 Corp). The error (Relative Difference) is defined below.

$$\text{RelativeDifference} = \frac{\text{MarketPrice} - \text{ModelPrice}}{\text{ModelPrice}} \tag{8}$$

MarketPrice is the daily close price of the bond (obtained from Bloomberg), and ModelPrice is calculated by TF model pricer with S steps = 225, t steps = 6000.

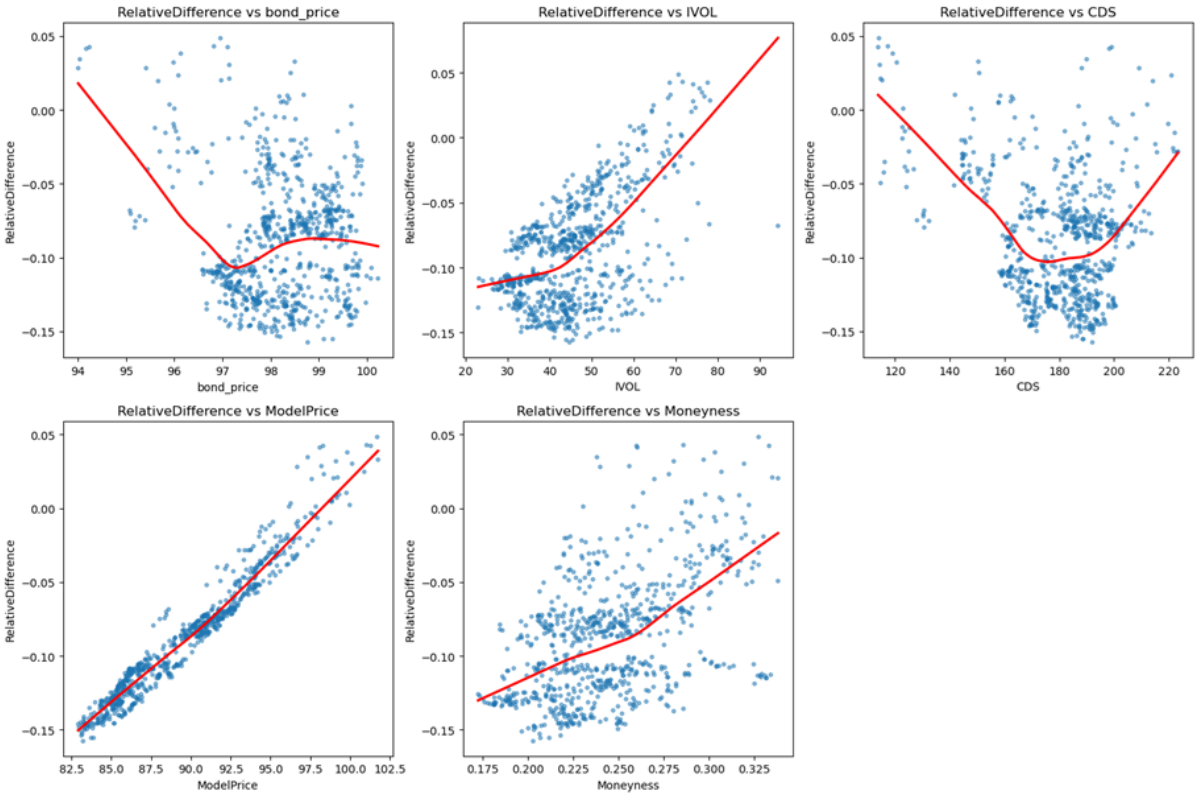


Figure 1: Scatter plot of Relative Difference across inputs

Scatter plots are first drawn as shown in Figure 1. These scatter plots show the relative difference across key inputs. It is observed that the relative difference in the model output was positively correlated with three inputs: implied volatility (IVOL), moneyness, and model price. Also, the correlation between the relative difference and

model price was particularly strong, with a coefficient of 0.97, indicating a near-linear relationship. Finally, the relative difference in model price has a "U-shape" distribution in CDS, which indicates overestimation on both sides of CDS input.

This analysis raised questions about the model's sensitivity to certain inputs and highlighted areas for potential improvement.

4.3 Suggested Improvements

Based on the error analysis, several avenues for improving the TF model were identified. First, a proportional bias was observed due to the linear relationship between the relative difference and the model price. To address this, a linear regression adjustment could be applied to correct this component of the error. Second, dynamic implied volatility (IV) adjustment is needed. Previous approaches applied a fixed ratio derived from the calibrated best volatility over the original IV data, which improved performance but lacked theoretical justification. A more dynamic adjustment would enhance both accuracy and interpretability.

The TF model implementation and error analysis revealed key insights into the model's behavior and identified areas for improvement. These improvements are discussed in section 5.

5 Exploration of Model Adjustment

According to the two patterns of error on different variables discovered, two approaches for adjusting the TF model are suggested.

5.1 Regression Adjustment for Systematic Proportional Bias

5.1.1 Adjustment Methodology

A systematic proportional bias was identified in the initial model pricing framework. This bias manifested as a consistent over- or under-prediction trend correlated with the magnitude of the model price. To address this, a regression-based adjustment was implemented.

Taking one bond for example (TEVA 0 $\frac{1}{4}$ 02/01/26 Corp), the relative difference between market and model prices could be modeled as a linear function of the model price shown in equation 9 (every bond should have different regression expression).

$$\text{Estimated RelativeDifference} = -0.9782 + (0.0099 \times \text{ModelPrice}) \tag{9}$$

After regression result was attained, the regression coefficients could be used for estimating the relative difference and correct the systematic bias back. The calculation for the adjusted model price is shown in equation 11.

$$\text{AdjustedModelPrice} = \frac{\text{ModelPrice}}{\text{Estimated RelativeDifference} + 1} \tag{10}$$

5.1.2 Performance Improvement

The adjustment reduced the root mean squared error (RMSE) from 0.1010 to 0.0103, representing a **89.8% reduction in prediction error**. This is shown in Figure 2.

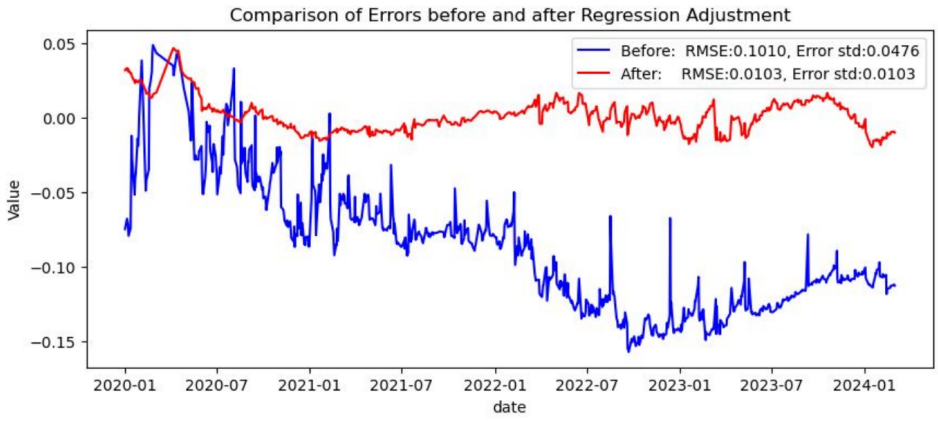


Figure 2: RMSE comparison before and after regression adjustment on ModelPrice

Due to the adjustment, the systematic over-prediction for high-value instruments is eliminated and the model outputs and market observations are better aligned. This demonstrates the efficacy of combining econometric regression with structural model corrections to mitigate inherent biases.

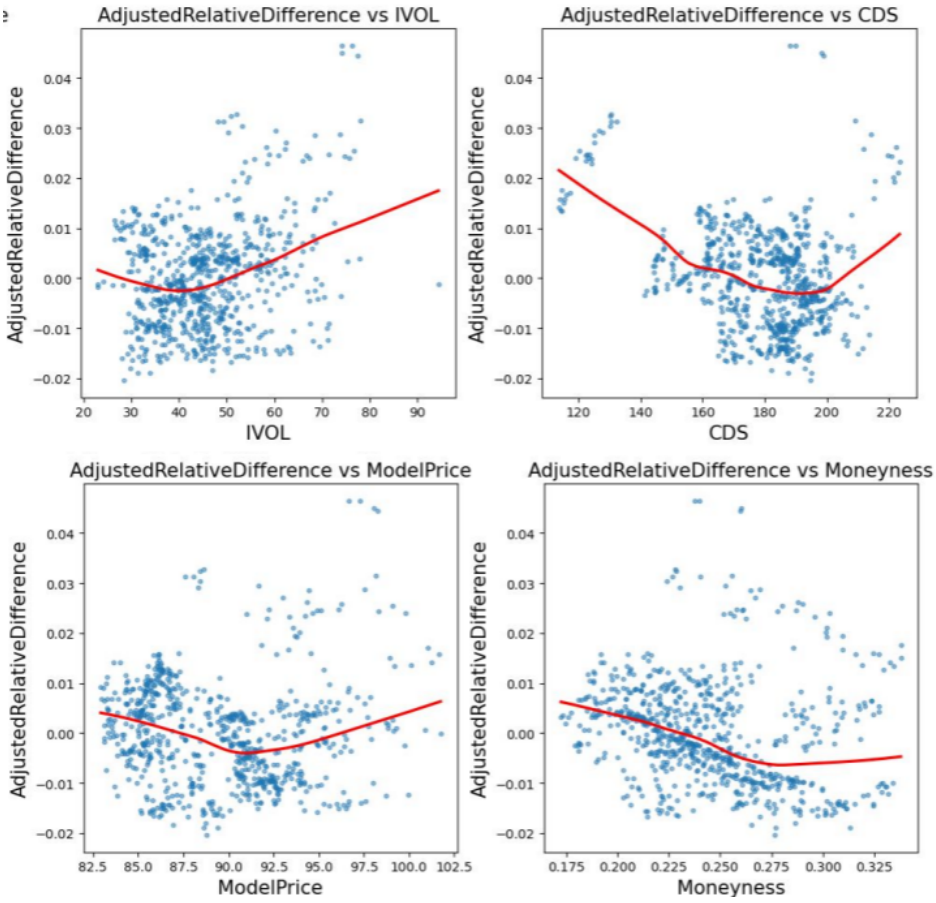


Figure 3: Adjusted Relative Difference across different input variables

Figure 3 shows the relative difference across the different variables after the adjustment was made. The results show that the linearity of error in model price was eliminated, but the skewness in Implied Volatility and U-shape in CDS still exists.

5.1.3 Universal Applicability

The results obtained on the single bond are then tested on all 85 bonds in the dataset to determine if this method is applicable for all bonds. Each regression is conducted on each bond to determine the adjusted model price. As shown in Figure 4, the out-of-sample RMSE after the adjustment is lower compared to the RMSE before the adjustment for all bonds. This indicates universal applicability and can be applied to all bonds.

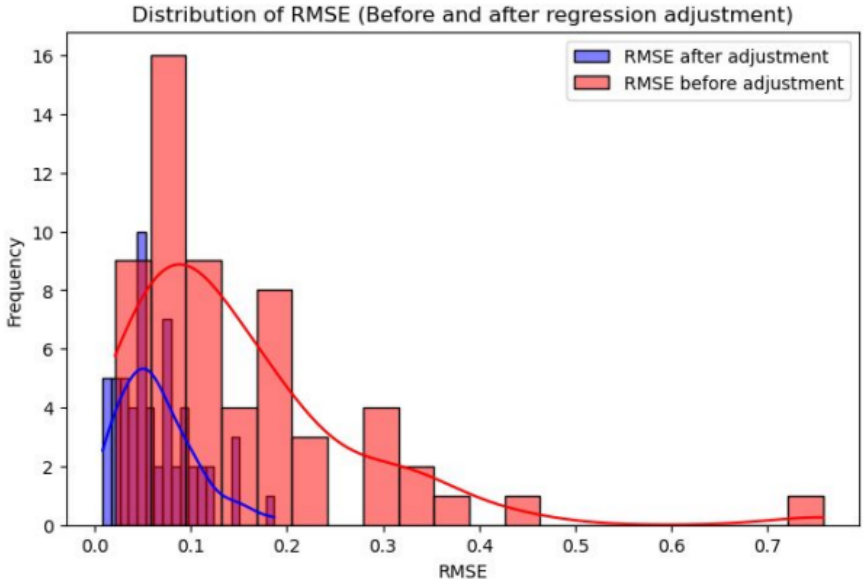


Figure 4: Distribution of RMSEs before and after regression adjustment

5.2 Implied Volatility Adjustment

After applying linear regression to address the proportional bias in the model price, the adjustment of the implied volatility input is examined to improve the accuracy of the TF model's prediction of the market price. Grid Search is first used to calibrate a series of Best Implied Volatilities (Best IVOL), which is the implied volatility input value that makes the model price closest to market price. The Adjustment Factor that is applied to the implied volatility is shown in equation 11. The goal is to find the best adjustment factor to make the difference between the model and market price as small as possible.

$$\text{Adjustment Factor} = \frac{\text{Best IVOL}}{\text{IVOL}} \tag{11}$$

A decision tree is tried first to model the relationship between the TF model's input variables and Adjustment Factor. A decision tree is used because of its perfect explainability and also categorical classification property. The decision tree for the implied volatility is large, and is included in the appendix.

A feature importance test is also run to determine which parameters are most important in determining the adjustment factor. Figure 5 shows that only interest rate, implied volatility, TTM and CDS are significant in the calculation of adjustment factor selection.

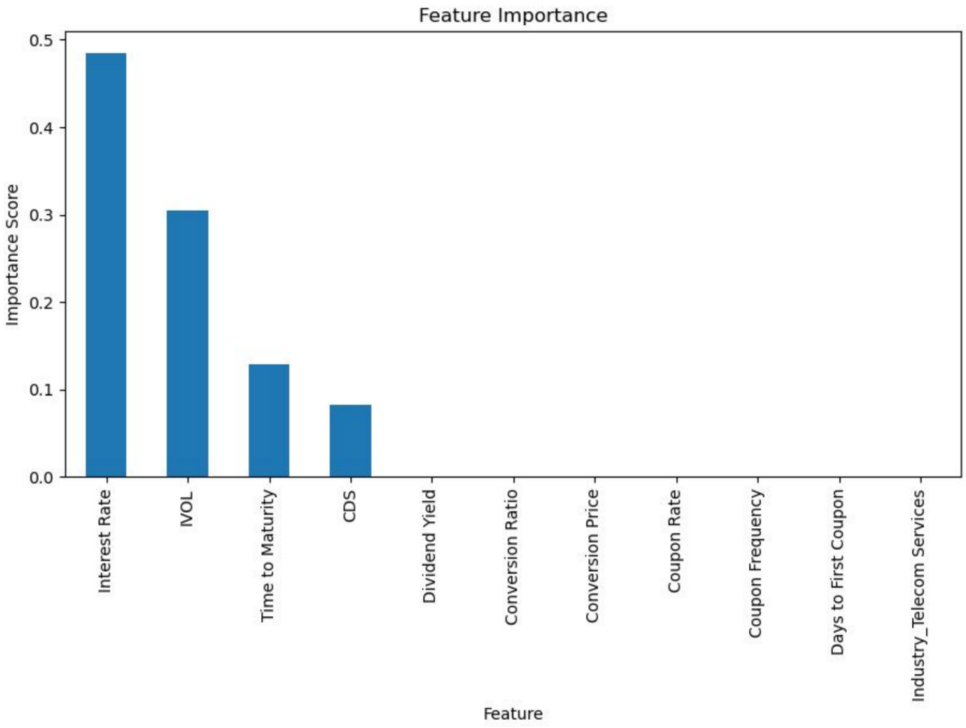


Figure 5: Feature importance of Decision Tree

Houlihan Lokey placed greater emphasis on using regressions to model implied volatility; therefore, the following section explores this approach in depth.

5.3 Regression-Based Input Adjustment Analysis

The goal of the regression analysis on implied volatility was to develop a robust method to estimate the calibrated volatility, derived by using grid search, based on the other TF model paramaters. The TF model parameters used within the regression were Moneyness, Time to Maturity (TTM), and Implied Volatility (IVOL). Both simple and logarithmic regression forms were tested, and each form included versions with and without an intercept.

Simple Intercepted model:

$$\text{Calibrated Volatility} = \beta_0 + \beta_1 \cdot \text{IVOL} + \beta_2 \cdot \text{Moneyness} + \beta_3 \cdot \text{TTM} \tag{12}$$

Simple Non-intercepted model:

$$\text{Calibrated Volatility} = \beta_0 \cdot \text{IVOL} + \beta_1 \cdot \text{Moneyness} + \beta_2 \cdot \text{TTM} \tag{13}$$

Logarithmic Intercepted model:

$$\log(\text{Calibrated Vol.}) = \beta_0 + \beta_1 \cdot \log(\text{IVOL}) + \beta_2 \cdot \log(\text{Moneyness}) + \beta_3 \cdot \log(\text{TTM}) \tag{14}$$

Logarithmic Non-intercepted model:

$$\log(\text{Calibrated Vol.}) = \beta_0 \cdot \log(\text{IVOL}) + \beta_1 \cdot \log(\text{Moneyness}) + \beta_2 \cdot \log(\text{TTM}) \tag{15}$$

For each regression, the dataset of 85 convertible bonds (47,577 data points) was split into an 80% training set and 20% test set for validation. Overall, linear regression models using simple and logarithmic transformations showed limited explanatory power, with relatively low R^2 values. Specifically, the simple intercepted model achieved an R^2 of approximately 0.175 on the test set, while the non-intercepted model performed significantly worse with an R^2 of just 0.0182. Similarly, the logarithmic transformation did not yield substantial improvements, achieving an R^2 of around 0.171 (intercepted) and 0.1588 (non-intercepted) respectively.

To combat this issue, the convertible bond data was first clustered by industry. Notable enhancements were observed when clustering by industry, especially within the REIT-Mortgage sector. In this specific industry segment, the test set demonstrated improved explanatory power with an R^2 of approximately 0.5223 (intercepted) and 0.5215 (non-intercepted). Correspondingly, the Mean Squared Errors (MSE) substantially reduced to 132.7916 (intercepted) and 133.0234 (non-intercepted), indicating enhanced predictive accuracy within this cluster.

To further improve the accuracy of the regression models, the data is also clustered separately by time to maturity and moneyness. This is done to see if the implied volatility regressions on specific ranges of the different parameters have increased accuracy compared to unified parameters. Table 1 shows the clustered ranges for time to maturity and Table 2 shows the clustered ranges for moneyness.

Table 1: Time to Maturity Clusters

Time to Maturity (years)
(0, 1)
[1, 2)
[2, 3)
[3, 4)
[4, 5)
[5, 6)
[6, 7)
[7, 10)
[10, ∞)

Table 2: Moneyness Clusters

Moneyness	**S Range**
Far In The Money (Far ITM)	$S > 1.5 \cdot CP$
Close In The Money (Close ITM)	$1.1 \cdot CP < S < 1.5 \cdot CP$
At The Money (ATM)	$0.9 \cdot CP < S < 1.1 \cdot CP$
Close Out The Money (Close OTM)	$0.5 \cdot CP < S < 0.9 \cdot CP$
Far Out The Money (Far OTM)	$S < 0.5 \cdot CP$

Unfortunately, both approaches (clustering by time to maturity or moneyness) had negligable improvements. In both the predictive models with an intercept and without an intercept, the test sets exhibited negative R^2 values, indicating that the linear regression was unable to explain the variance in the target variable for these clusters. Furthermore, while the mean squared error (MSE) for the simple linear model was around 140, and for the logarithmic model was approximately 0.15, these values do not necessarily signal good

predictive performance. Rather, they highlight that the models may be failing to capture the underlying relationships or may be overfitting. Consequently, clustering convertible bonds by moneyness or time to maturity alone does not appear to yield meaningful improvements in predictive accuracy, suggesting that more sophisticated techniques or additional explanatory variables may be necessary to accurately model convertible bond pricing.

However, clustering by time to maturity and industry revealed pronounced improvements. The REIT-Mortgage subset, achieved a high test-set R^2 of approximately 0.994 with an extremely low MSE of 0.17 (intercepted). Similarly, clustering by moneyness and industry produced similar predictive enhancements. The REIT-Mortgage subset achieved a R^2 value of approximately 0.99 and an exceptionally low MSE of 0.002.

Tables 3, 4, and 5 contain the result of our regression models. Note that Table 4 and Table 5 are the average accuracy metrics across all ranges.

Table 3: Performance of general regression model

Test Set	**R^2 (intercepted)**	**MSE (intercepted)**	**R^2 (non-intercepted)**	**MSE (non-intercepted)**
Simple	0.1750	339.7146	0.0182	404.2654
Log	0.1710	341.3340	0.1588	346.3732
REIT-mortgage	0.5223	132.7916	0.5215	133.0234

Table 4: Performance of Clustering by Time to Maturity

Test Set (20%)	**MSE (intercepted)**	**R^2 (intercepted)**	**MSE (non-intercepted)**	**R^2 (non-intercepted)**
Simple	113.75	-3.976	128.86	-1.117
Log	0.41	-0.503	0.45	-0.926
REIT-Mortgage (simple)	0.17	0.994	0.12	0.997

Table 5: Performance of Clustering by Moneyness

Test Set (20%)	**MSE (intercepted)**	**R^2 (intercepted)**	**MSE (non-intercepted)**	**R^2 (non-intercepted)**
Simple	122.78	-0.07	145.46	-0.26
Log	0.25	-0.211	0.27	-0.371
REIT-Mortgage (simple)	0.002	0.99	0.003	0.99

It should be noted that as the convertible bond becomes more out-the-money, the accuracy of the regression model declines. This is due to the fact that as the convertible bond behaves more as a debt instrument, the implied volatility, which is derived from an option pricing model, becomes more difficult to accurately estimate.

Clustering the convertible bond data by industry yielded improved accuracy on average across all industries for the implied volatility regressions. Based on these results, when using regression to adjust the implied volatility to increase the accuracy of the TF model, it is important to cluster the data by either time to maturity or moneyness, as well as by industry. However, while these targeted clustering strategies significantly improved model accuracy, the overall predictive performance of simple linear regression remains constrained. This limitation highlights the necessity for more sophisticated, non-linear modeling approaches or the inclusion of additional explanatory variables to fully capture complex market dynamics and achieve robust convertible bond pricing predictions.

6 Data Synthesis for TF Model Replication

To train a machine-learning surrogate of the TF model, it is necessary to generate far more data than the original 47,577 market observations. Accordingly, parameter vectors are synthesized by first fitting each model input—volatility, credit spread, time to maturity, etc.—to its empirical distribution derived from an 85-bond sample, and then imposing the observed cross-parameter correlations via a multivariate sampling procedure. Applying the Crank–Nicolson PDE solver to these ten million synthetic parameter sets yields ten million corresponding "synthetic" TF prices. These prices, produced under realistic joint-market behavior, serve as high-quality targets for training and validating the machine-learning replication.

6.1 Data Preparation for Machine Learning

Using the original dataset of the 85 convertible bonds, correlations between the different TF model variables were found. The correlations are reflected in the following correlation matrix.

Table 6: Correlation matrix between TF model parameters

	CDS	Best IVOL	Stock Price	Bond Price	Interest Rate
CDS	1	0.18	−0.11	−0.08	0.19
Best IVOL	0.18	1	−0.10	0.31	0.17
Stock Price	−0.11	−0.10	1	0.11	0.00
Bond Price	−0.08	0.31	0.11	1	−0.05
Interest Rate	0.19	0.17	0.00	−0.05	1

In addition, statistical significance tests were also performed to determine which relationships between the parameters were meaningful. The correlations between the different TF model parameters were all significant, with the exception of interest rate and stock price (as the p-value =0.560). These correlations were useful when constructing accurate synthetic training and test data sets that represent real-world market data for replication of the TF model using machine learning. This replication will be discussed in future sections.

To construct robust synthetic data for convertible bond pricing, analyzing the quantiles on our existing parameters dataset to identify realistic parameter ranges. The results of this analysis are in the appendix (Figure 18 in appendix). This statistical approach captures the distribution and variability inherent in market data, ensuring our synthetic scenarios reflected actual market conditions.

Based on the quantile analysis, the following parameter ranges were established for generating synthetic data:

These parameters, carefully calibrated using real market quantiles, enable more accurate and representative synthetic data generation, critical for evaluating and enhancing convertible bond valuation models. Using synthetic parameter ranges derived from our quantile analysis, (Table 7), the TF PDEs are numerically solved via finite difference methods to generate target convertible bond prices (V).

Table 7: Established Parameter Ranges for Synthetic Data

Parameter	**Range**
Bond Price	\$50 – \$150
Implied Volatility (IVOL)	5% – 50%
Credit Default Swap (CDS) Spreads	40 bps – 400 bps
Underlying Stock Price (S)	\$10 – \$100
Interest Rate (R)	1% – 5%
Time to Maturity (TtM)	1 – 10 years
Coupon Rate (Cr)	0.3% – 6%
Coupon Frequency	80% semiannual, 20% quarterly
Conversion Ratio	5 – 100
Conversion Price	\$15 – \$175
Dividend Yield	30% at 0%, 70% uniformly distributed (0% – 17.5%)

6.2 Crank–Nicolson Finite–Difference Scheme for the TF PDEs

6.2.1 Crank–Nicolson Discretisation

The Tsiveriotis–Fernandes (TF) system consists of two coupled parabolic partial differential equations—one for the cash (bond) component $B(S,t)$ and one for the total convertible-bond value $V(S,t)$. A second-order, unconditionally stable Crank–Nicolson (CN) scheme is adopted to advance the solution in time and to supply high-fidelity labels for subsequent machine-learning experiments. Table 8 reflects the parameter definitions used in this analysis.

Table 8: Model Parameters and Descriptions

Parameter	**Description**
σ	Annualised volatility of the underlying stock price
r	Continuously compounded risk-free rate
q	Continuous dividend yield
s	Credit spread applied to the cash component
$S_{\max}$	Upper boundary of the spatial grid
M	Number of spatial intervals; $\Delta S = S_{\max}/M$
T	Time to maturity of the bond
N	Number of temporal steps; $\Delta t = T/N$
h	Shorthand for $\Delta t/2$, used in the Crank-Nicolson stencils

Spatial nodes are $S_i = i\,\Delta S$ for $i = 0, \ldots, M$ and temporal nodes are $t_n = n\,\Delta t$ for $n = 0, \ldots, N$. Discrete values are denoted $B_i^n = B(S_i, t_n)$ and $V_i^n = V(S_i, t_n)$.
Central differences approximate the first and second spatial derivatives,

$$D_S f_i^n = \frac{f_{i+1}^n - f_{i-1}^n}{2\,\Delta S}, \qquad D_S^2 f_i^n = \frac{f_{i+1}^n - 2f_i^n + f_{i-1}^n}{\Delta S^2},$$

while the temporal average $\langle f \rangle_i^{n+\frac{1}{2}} = \frac{1}{2}(f_i^{n+1} + f_i^n)$ appears in the CN formulation.

For every interior index $i = 1, \ldots, M-1$,

$$D_i = \tfrac{1}{2}\,\sigma^2 S_i^2, \qquad a_i = \frac{D_i}{\Delta S^2} - \frac{(r-q)S_i}{2\,\Delta S}, \qquad c_i = \frac{D_i}{\Delta S^2} + \frac{(r-q)S_i}{2\,\Delta S},$$

$$b_i^B = -\frac{2D_i}{\Delta S^2} - (r+s), \qquad b_i^V = -\frac{2D_i}{\Delta S^2} - r.$$

Parameters a_i and c_i weight the diffusive and convective terms; b_i^B and b_i^V capture discounting and credit effects. Applying CN to the bond-only equation produces, for $i = 1, \ldots, M-1$,

$$-h\,a_i\,B_{i-1}^{n+1} + \big(1 - h\,b_i^B\big)B_i^{n+1} - h\,c_i\,B_{i+1}^{n+1} = h\,a_i\,B_{i-1}^{n} + \big(1 + h\,b_i^B\big)B_i^{n} + h\,c_i\,B_{i+1}^{n}. \quad (16)$$

The left-hand side contains the unknown time-level $n+1$; the right-hand side is fully known from level n. For Total value discretisation, the second stencil couples V and B:

$$\begin{aligned} &- h\,a_i\,V_{i-1}^{n+1} + \big(1 - h\,b_i^V\big)V_i^{n+1} - h\,c_i\,V_{i+1}^{n+1} + h\,r\,B_i^{n+1} \\ &= h\,a_i\,V_{i-1}^{n} + \big(1 + h\,b_i^V\big)V_i^{n} + h\,c_i\,V_{i+1}^{n} - h\,r\,B_i^{n}. \end{aligned} \quad (17)$$

The term $h\,r\,B$ transfers credit-adjusted coupon information from the bond component into the total value equation. Stacking equations (16)–(17) over all interior nodes yields a block-tridiagonal linear system

$$\mathcal{A}\,\mathbf{X}^{n+1} = \mathbf{d}(\mathbf{X}^n), \qquad \mathbf{X}_i = \begin{bmatrix} B_i \\ V_i \end{bmatrix},$$

which is solved at every time step by a block Thomas algorithm. Appropriate terminal pay-off conditions at $t = T$ and boundary conditions at $S = 0$ and $S = S_{\max}$ close the system, ensuring stability and second-order accuracy throughout the grid.

6.3 Stability and Convergence Analysis

6.3.1 Stability Guideline (CFL Condition)

Although the Crank–Nicolson (CN) finite-difference scheme is unconditionally stable, the Courant–Friedrichs–Lewy (CFL) condition could be adopted as a practical guideline when choosing the spatial step size Δx and temporal step size Δt. In particular, a CFL-type ratio could be enforced

$$\frac{\sigma^2\,\Delta t}{(\Delta x)^2} \lesssim C_{\max},$$

where $C_{\max}$ is a safety factor (e.g. 0.5–1.0) borrowed from the explicit scheme. By linking Δx and Δt through this ratio, it could be ensured that our grid refinements preserve both stability and consistency as refining the mesh.

6.3.2 Convergence Criterion

To quantify convergence, suppose *convergence coefficient* ε_n at refinement level n as the average relative change over the last three model prices:

$$\varepsilon_n \;=\; \frac{1}{3}\sum_{k=0}^{2}\left|\frac{V_{n-k} - V_{n-k-1}}{V_{n-k-1}}\right| \times 100\%,$$

where V_i denotes the option value computed on the ith refinement. the solution could be deemed converged once $\varepsilon_n < 0.1\%$.

6.3.3 Numerical Experiment Setup

In the numerical experiments, the mesh was refined by halving both Δx and Δt at each level, while the CFL-guided ratio was maintained. All computed prices were compared against a high-resolution "benchmark" solution obtained on a very fine mesh. Throughout these runs, three metrics were tracked: the model price at maturity, $V(T)$; the computation time (in CPU seconds); and the convergence coefficient, ε_n.

6.3.4 Results and Discussion

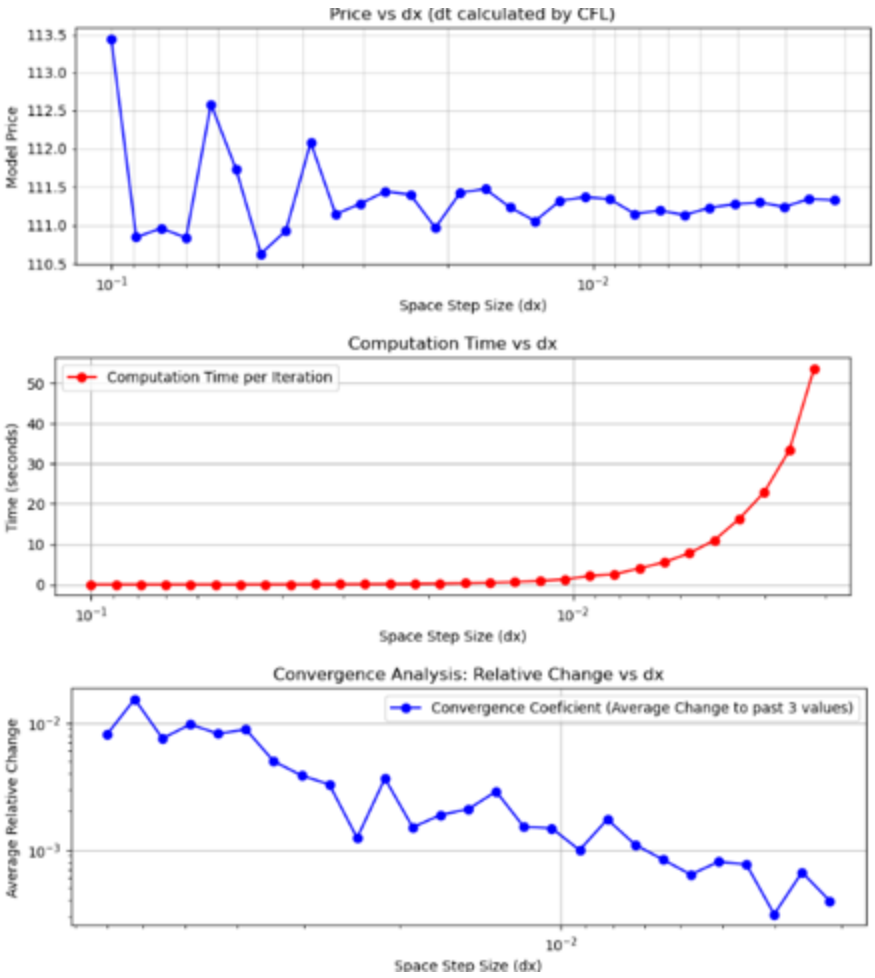


Figure 6: Convergence coefficient, TF model price and computation time as functions of Δx and Δt.

As the mesh was refined, the computed option price was observed to approach the benchmark monotonically, with diminishing increments between successive refinements. At the same time, CPU time was found to increase by roughly a factor of four each time Δx and Δt were halved, reflecting the quadratic growth in grid points. The convergence coefficient ε_n declined below the 0.1% threshold after five refinements, despite minor fluctuations at intermediate levels. Monotonic convergence of the Crank–Nicolson scheme, guided by the CFL refinement strategy, is confirmed by these observations, and a transparent trade-off between accuracy and computational expense is illustrated.

6.3.5 Application to Synthetic Data Generation

Having identified that $(\Delta x^*, \Delta t^*)$, the refinement level at which $\varepsilon_n < 0.1\%$, offers sufficient accuracy at reasonable cost, $(\Delta x^*, \Delta t^*)$ for all subsequent model runs were adopt. This ensures that the synthetic dataset—comprising several thousand model-priced trajectories—reflects converged CN solutions, thus providing a high-quality foundation for training our machine-learning replicator models.

7 Supervised Machine Learning for Convertible Bond Pricing

7.1 Overview

Machine learning (ML) for replication attempts to reproduce the "fair-value" or conversion decisions indicated by a more fundamental model. The Tsiveriotis-Fernandes (TF) partial differential equation (PDE) approach for pricing convertible bonds is leveraged, in which the outputs (i.e., theoretical bond value or when to convert) are treated as a reference ground truth. Below, the ML branch of supervised learning is outlined, in which a Neural Network Model (MLP) is employed. Random forest regressor (RFR), and Gradient Boosting (GBR) models were also explored, but their results were not as strong as the Neural Network models.

7.2 Random Forest Regressor

A random forest regressor model (RFR) was constructed for convertible bond pricing and tuned using GridSearchCV. The following table shows the hyperparameters and the different values that were used to conduct the grid search.

Table 9: Hyperparameter values used in the RFR model

Hyperparameter	**Values**
Number of estimators	100, 300, 500
Max depth	5, 10, 20, None
Max features	sqrt, log2, auto

The following hyperparameters were selected to optimize the random forest regressor model (RFR): Number of estimators (500), max depth (none), max features (sqrt). The final model achieved an RMSE of approximately 11.8754 and an R^2 of about 0.9778 on the test set. Additionaly, the mean absolute percentage error (MAPE) was 34.93%. This percentage error equated to around a 20 dollar error on the average convertible bond. Training took around 0.3875 hours, while predictions were generated in about 3.47 seconds. Note that training time and prediction speeds may vary depending on computational power.

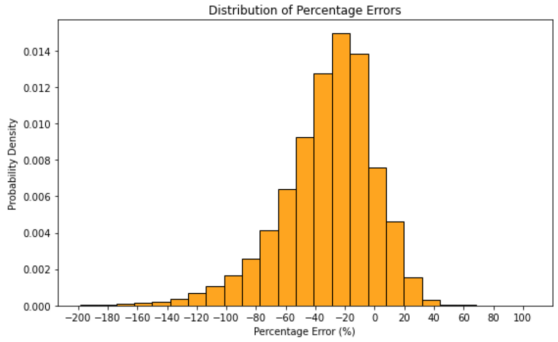


Figure 7: Histogram of Percent Errors

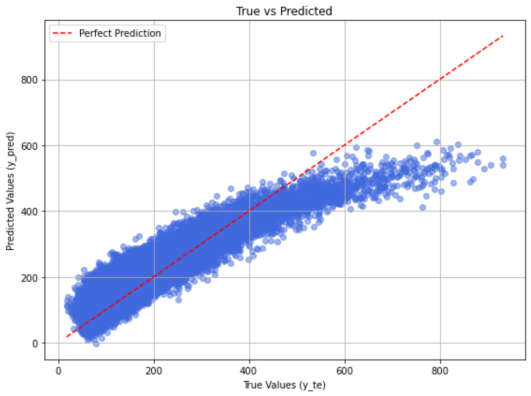


Figure 8: Prediction vs. Actual Prices

Based on the above figures, a majority of the data points lie below the red dotted line, which represents a perfect prediction. The random forest regressor model (RFR) therefore underprices the convertible bond on average, with higher prices deviating more from a perfect fit. In addition, the histogram shows a wide disparity of percentage errors, highlighting the poor performance of the model. Although, this is a lower Mean Absolute Percentage Error compared to a Neural Network with the same training set size, the Random Forest Regressor is not explored further as the Neural Network performs better with larger training sets.

7.3 Gradient Boosting

A Gradient Boosting Model (GBR) was constructed for convertible bond pricing and tuned using GridSearchCV. The following table shows the hyperparameters and the different values that were used to conduct the grid search.

Table 10: Hyperparameter values used in the GBR model

Hyperparameter	**Values**
Number of estimators	100, 300, 500
Max depth	5, 10, 20, None
Learning Rate	0.01, 0.05, 0.1
Subsample	0.8, 1.0

The following hyperparameters were selected to optimize the Gradient Boosting model (GBR): Number of estimators (500), max depth (none), learning rate (0.01), and subsample (1.0). The final model achieved an RMSE of approximately 7.9919 and an R^2 of about 0.9889 on the test set. The model achieved a mean absolute percentage error (MAPE) of 25.46%, equating dollar errors around $15-20 per bond. Training took around 2.84 hours, while predictions were generated in about 0.0955 seconds. Note that training time and prediction speeds may vary depending on computational power.

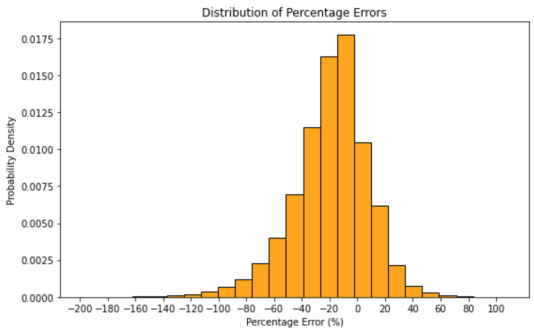


Figure 9: Histogram of Percent Errors

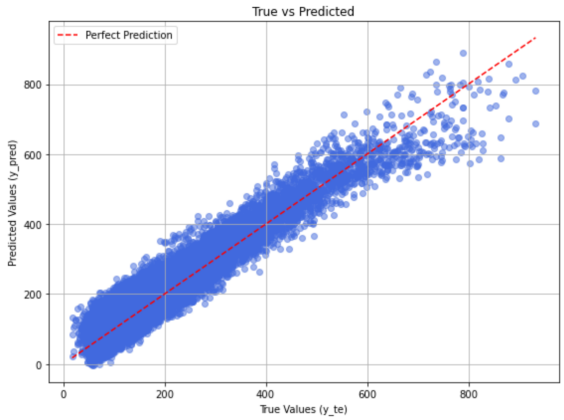


Figure 10: Prediction vs. Actual Prices

Based on the above figure, the gradient boosting model (GBR) slightly fixes the underpricing issue that the random forest regressor model (RFR) had. The histogram for GBR still shows a wide disparity of percentage errors, highlighting its poor performance as well. Although, the Gradient Boosting model had a lower Mean Absolute Percentage Error compared to a both the Random Forest and Neural Network with the same training set size, the Gradient Boosting model is not explored further, as similarly to the random forest regressor, the Neural Network performs better with larger training sets.

7.4 Neural Networks

Several feed-forward MLP with 2–3 hidden layers (depending on the chosen architecture) were trained with varying training set size for convertible bond pricing and tuned using GridSearchCV with the different hyperparameters listed in the below table.

Table 11: Hyperparameter values used in the MLP model

Hyperparameter	**Values**
Hidden Layer Sizes	(50,), (100,), (50,50), (100,50)
Activation Function	ReLU, tanh
Solver	Adam, SGD
Alpha (Regularization)	0.0001, 0.001, 0.01
Learning Rate Init	0.0001, 0.001, 0.01
Max Iterations	2000

Each Neural Network was trained on a different sample size dataset (40 thousand, 400 thousand, 4 million, and 10 million) with a 80/20 train—test split. Mean Squared Error (MSE) was used as the loss function and an appropriate optimizer based on the selected solver (Adam or SGD). Each configuration was run to convergence (up to 2000 iterations), with the best performing combination being selected based on the lowest validation RMSE

7.4.1 40k Neural Network

For the 40k training set, the final model achieved an RMSE of 7.25 and an R^2 of 0.9917 on the test set. Training took 1 minute, while predictions were generated in

2.86 microseconds per prediction, demonstrating high efficiency in prediction speed. The accuracy, while higher than the Random forest and XGBoost models, was still very low with a mean absolute percentage error of 108.6%. The distribution of percentage errors is shown in the below figure.

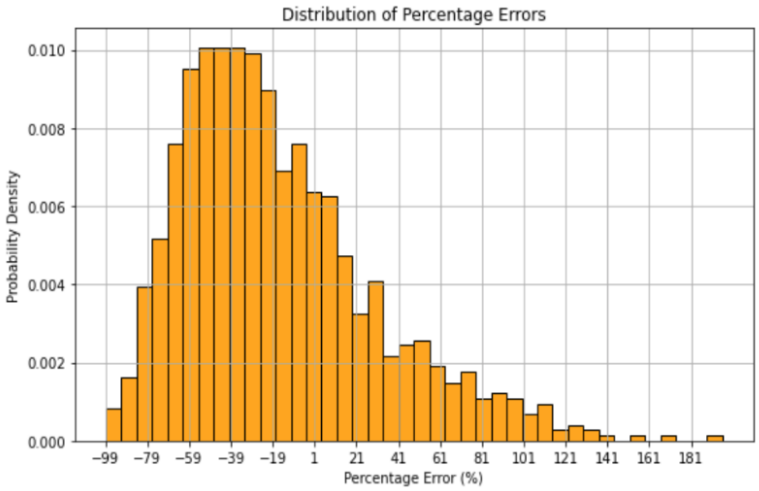


Figure 11: 40k Neural Network Percentage Errors

The distribution of errors was very wide with only 0.8% of errors being under the 1% threshold. This equated to dollar errors of -$123 at the 10% dollar error quantile (prediction – test value) and $38 at the 90% dollar error quantile. These large errors are mostly due to too small of a sample size for training, so the sample size was increased.

7.4.2 400k Neural Network

For the 400k training set, the final model achieved an RMSE of 2.80 and an R^2 of 0.9987 on the test set. Training took 12.2 minutes, while predictions were generated in 3.6 microseconds per prediction. The accuracy clearly improved from the 40k dataset neural network with a new mean absolute percentage error (MAPE) of 1.11%. The distribution of percentage errors for the 400k dataset neural network is shown in the below figure.

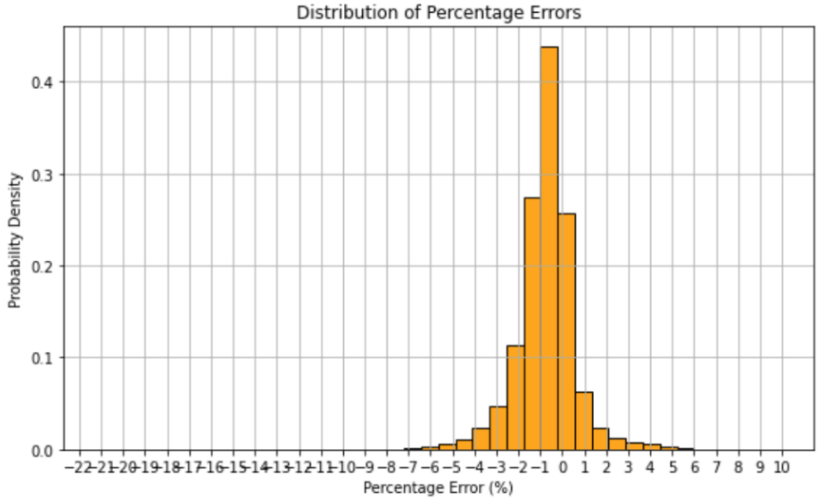


Figure 12: 400k Neural Network Percentage Errors

52.87% of predictions were within the 1% threshold, which was much higher than 0.8% for the 40k dataset neural network. The dollar errors at the 10% quantile was -$2.19 and $0.50 at the 90% dollar error quantile. A slight underprediction bias of the neural network was observed, as more errors underpredicted the TF model price compared to overprediction. This was also observed in the 40k dataset neural network, but is most likely due to sample bias. To further improve the accuracy of the neural network, the dataset size is once again increased.

7.4.3 4 Million Neural Network

For the 4 million training set, the final model achieved an RMSE of 3.16 and an R^2 of 0.9993 on the test set. Training took 2.86 hours, while predictions were generated in 2.04 microseconds per prediction. The accuracy slightly improved from the 400k dataset neural network with a new mean absolute percentage error (MAPE) of 0.733%. This metric is within the 1% threshold, and therefore the Neural Network model has a high enough accuracy to be employed. The distribution of percentage errors for the 4 Million dataset neural network is shown in the below figure.

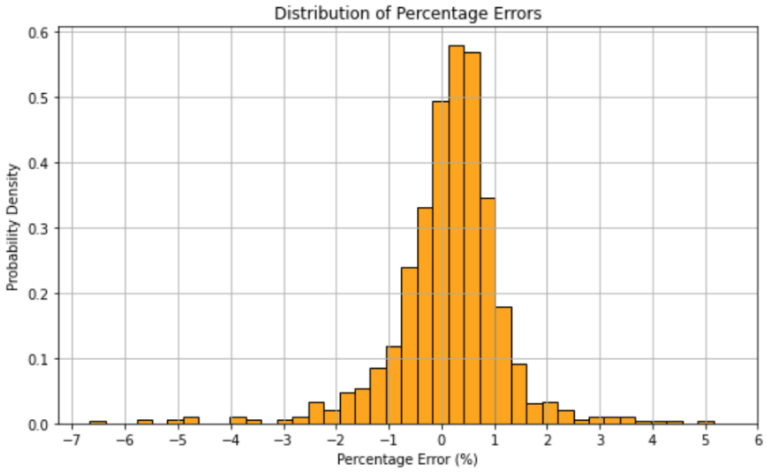


Figure 13: 4 Million Neural Network Percentage Errors

78.10% of predictions were within the 1% threshold and the dollar error at the 10% quantile was -$1.29 and $1.55 at the 90% dollar error quantile. The underprediction bias of the neural network is now much smaller, as the errors are much more symmetrical. After training this Neural Network model, the dataset size was increased once again to 10 million. This did not yield improved results, and future work can be done to determine if the 4 million sized dataset contains all the relevant data to train the most accurate Neural Network model.

7.4.4 Neural Network Feature Importance

The features used to train the Neural Network model are listed in the table below. A permutation importance test was run to determine which features were the most important in the model. It was determined that moneyness, bond price, and conversion value were the three features that mainly determined the price of the convertible bond, while interest rate, dividend yield, and the credit default swap affected the price the least.

Table 12: TF model Neural Network Features

Feature
Time (T)
Bond Price (Pr)
IVOL
Stock Price (S)
Moneyness = S/cv
Coupon Rate (cp)
Conversion Price (cv)
Dividend Rate (d)
Interest Rate (r)
Conversion Premium = $Pr - c$

8 Reinforcement Learning Approach

8.1 Motivation

Reinforcement learning (RL) tackles sequential decision problems by learning a policy $\pi(a \mid s)$ that maximises expected cumulative reward in a Markov decision process (MDP). Because early-exercise instruments, such as American options and convertible bonds (CBs) are naturally phrased as optimal-stopping MDPs, RL provides a flexible alternative to deterministic solvers. We first validate our methods on American options—where binomial trees and Longstaff–Schwartz (LSM) offer reliable benchmarks—before extending the same machinery to the Tsiveriotis–Fernandes (TF) PDE framework used for CBs.

8.2 MDP Formulation for Convertible Bonds

The State, s_t gathers observable market variables plus the TF continuation price.

$$s_t = \big(S_t,\ \text{ttm},\ \text{IVOL},\ \text{CDS},\ r_t,\ \text{TF cont. value}\big)$$

The actions are either a binary hold/convert choice or a continuous PDE-price guess, depending on the algorithm. The reward, R_t is listed as follows.

$$R_t = \begin{cases} -(\text{PDE} - \hat{P})^2, & \text{price-matching,} \\ \text{conversion payoff} - \text{continuation value}, & \text{exercise.} \end{cases}$$

Stock paths follow GBM; the TF engine updates continuation values each step.

8.3 Algorithms Evaluated

The following algorithms are evaluated: Q-Learning serves as a fast and interpretable baseline but tends to struggle in high-dimensional state grids. Least-Squares Policy Iteration (LSPI) is a batch off-policy method that is significantly more sample-efficient than traditional tabular updates. Policy Gradient methods such as REINFORCE employ neural policies to handle continuous spaces, though they often require techniques like variance reduction and reward scaling for stability. Finally, Deep Deterministic Policy

Gradient (DDPG) uses an actor–critic framework suited for continuous price predictions; while powerful, it is notably sensitive to hyper-parameter choices.

Coarse grid-search over $\alpha \in [5\times10^{-4},\ 5\times10^{-2}]$ and $\gamma \in [0.95,\ 0.99]$ showed that clipping both rewards and action ranges (e.g. PDE $\in [50, 300]$) materially improved convergence.

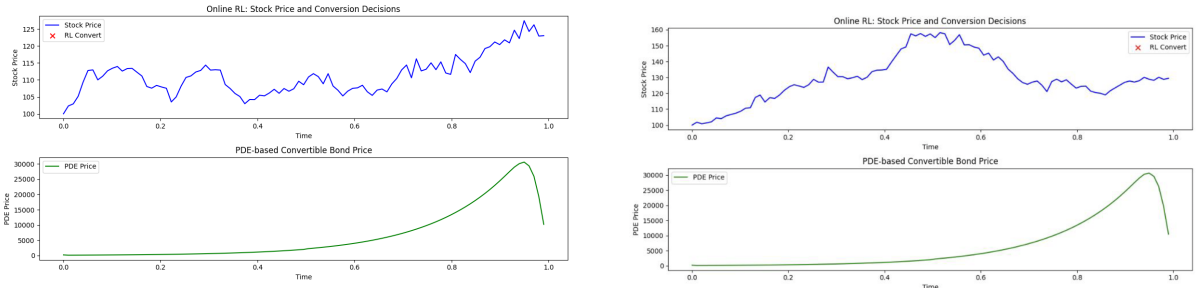
8.4 Training Setup

Synthetic datasets (40k – 300k rows) feed online agents and offline batches. Each episode traces a bond from issuance to maturity/conversion. Convergence is monitored via MSE to TF prices and policy stabilization.

All agents were trained on a synthetic convertible-bond dataset in which the underlying stock follows a calibrated geometric-Brownian-motion path and credit spreads evolve stochastically within empirically observed bounds. For the online experiments the entire bond lifecycles end-to-end were simulated, updating the policy after each episode; Policy-Gradient agents were run for 40,000, 100,000 and 300,000 episodes, using a learning rate of 1×10^{-3}, discount factor $\gamma = 0.99$, and a single 64-unit hidden layer. For the offline comparison a random 500,000-row subset was drawn of the full 4Mil transition corpus and trained tabular Q-Learning with $\alpha = 0.05$, $\gamma = 0.99$, and ϵ-greedy exploration annealed from 1.0 to 0.1. DDPG used identical state features but produced a continuous price guess, with actor/critic learning rates of 1×10^{-4} and target-network soft-updates $\tau = 0.001$. Mean-squared error (MSE) to the Tsiveriotis–Fernandes (TF) benchmark was recorded on an unseen validation set every 5 000 steps.

8.5 CB Pricing Replication Results

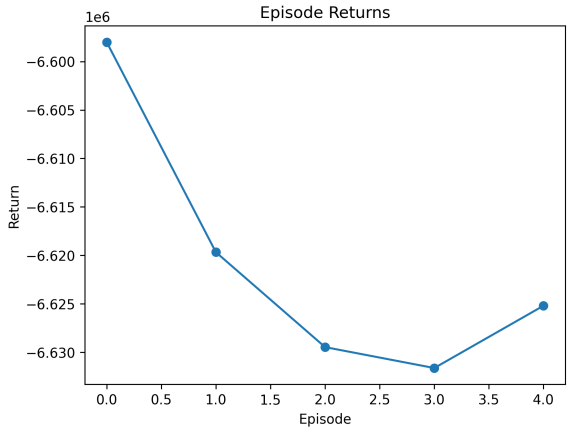
Early (40 k) Policy-Gradient training produced sporadic conversions and large errors (Figure 14a); extending to 100 k–300 k episodes aligned the policy with the TF benchmark and cut MSE by an order of magnitude (Figures 14b, 15a). Offline Q-Learning on a 500 k subsample achieved the lowest MSE (0.62), while DDPG delivered the most accurate continuous pricing once stabilised.



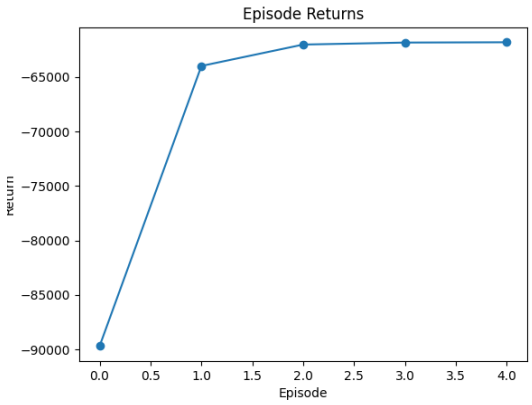
(a) Online RL vs. PDE after 40k episodes.

(b) Online RL vs. PDE after 100k episodes.

Figure 14: Comparison of Online RL and PDE performance at different training durations.



(a) Policy-Gradient reward trajectory.



(b) Q-Learning cumulative returns (100k episodes).

Figure 15: Comparison of learning curves for different RL methods.

Figure 14 contrasts Policy-Gradient performance after 40k and 100k episodes. After 40k iterations (left panel) the agent's conversion tags (red X's) scatter widely around the TF price path (blue), indicating premature or delayed exercises and yielding high MSE. By 100k episodes (right panel) the conversion markers cluster tightly near the late-stage TF price surge, showing that the policy has learned the optimal stopping frontier and cutting the error by roughly one order of magnitude. Figure 15 shows learning trajectories: the left sub-figure plots the steadily rising cumulative reward of Policy-Gradient, while the right sub-figure depicts Q-Learning's rapid early ascent and plateau after 100k updates. The flatter tail of Q-Learning reflects discrete-grid saturation, whereas the smoother but slower Policy-Gradient curve continues to gain as it fine-tunes a continuous policy surface. Together the plots corroborate the numerical results in the text: offline Q-Learning attains the lowest MSE (0.62) on the subsample, but DDPG ultimately yields the most accurate continuous pricing once its actor–critic pair has stabilised.

8.6 Quantitative Comparison

Three controlled passes—identical except for PDE-guess range—appear in Table 13. Constraining actions to $[50, 300]$ cut MSE by two orders of magnitude and raised R^2 from < -10 to ≈ -0.5.

Table 13: Quantitative Comparison of Three Passes (mean over two seeds)

Pass	**PDE Range**	**MSE (Train)**	**MSE (Test)**	**R^2 (Test)**	**Key Takeaway**
1	$[1,\ 10^7]$	$\mathcal{O}(10^8)$	Unstable	< -10	Too-wide action space.
2	Heuristic	1.8×10^4	8.6×10^4	$[-13, -2]$	Fewer outliers, still mis-scaled.
3	$[50,\ 300]$	9.3×10^3	9.3×10^3	≈ -0.5	Realistic range stabilises training.

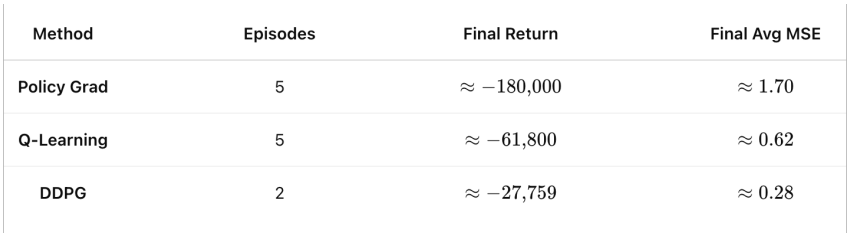
8.7 Practical Insights and Limitations

Several practical insights and limitations emerged during experimentation. One key issue is the presence of numerical noise in the terminal condition of the PDE, which tends to introduce price spikes near maturity. This can be partially mitigated through techniques such as reward clipping and extending the training duration. Another challenge lies in reward sparsity, as the learning signal is concentrated entirely in the terminal payoff. To address this, reward shaping using continuation values proved essential in guiding learning. On the modeling side, the use of geometric Brownian motion (GBM) limits data realism. More complex dynamics—such as jump-diffusion processes and stochastic spread models—are planned as future improvements. Finally, computational cost remains a significant bottleneck: even with GPU-accelerated inference, training requires millions of environment steps, most of which are constrained by CPU performance.

8.8 Algorithm Summary

Table 14: Summary of Algorithms Evaluated

Algorithm	**Motivation**	**Core Params**	**Key Findings**
Q-Learning	Interpretable baseline	$\alpha = 0.05$, $\gamma = 0.99$	Rapid early gains; scales poorly in large grids.
Policy Gradient	Continuous actions	LR 10^{-3}, hidden 64	Smooth reward curve; high variance without clipping.
DDPG	Continuous price guess	Actor LR 10^{-4}, $\tau = 0.001$	Best continuous accuracy; sensitive to noise.



Method	Episodes	Final Return	Final Avg MSE
Policy Grad	5	$\approx -180{,}000$	≈ 1.70
Q-Learning	5	$\approx -61{,}800$	≈ 0.62
DDPG	2	$\approx -27{,}759$	≈ 0.28

Figure 16: Side-by-side algorithm comparison.

Table 14 presents a comparative summary of the reinforcement learning algorithms evaluated in this study. Q-Learning serves as a straightforward and interpretable baseline, demonstrating strong initial performance but limited scalability in larger state spaces. Policy Gradient methods were chosen for their suitability to continuous action spaces, yielding smoother reward trajectories but requiring techniques like reward clipping to manage high variance. DDPG, designed for continuous control tasks such as price prediction, achieved the highest accuracy among the tested algorithms but exhibited heightened sensitivity to environmental noise and hyperparameter tuning. Overall, each method displays distinct trade-offs between interpretability, stability, and performance under complex settings.

The table provides a quick side-by-side snapshot of how three reinforcement learning methods fared on the same task. Each row summarizes (i) the number of training episodes completed, (ii) the total return achieved, and (iii) the mean-squared error (MSE) of the final price predictions.

Policy Gradient required five episodes but ended with the largest loss (approximately –180k) and exhibited the poorest fit to the target price curve (MSE ≈ 1.70).

Q-learning also trained for five episodes, reduced that loss by roughly two-thirds (approximately –61.8k) and more than halved the prediction error (MSE ≈ 0.62).

DDPG (a continuous-action actor-critic method) outperformed both despite completing only two episodes: its return was the least negative (approximately –27.8k), and its predictions were closest to the ground truth (MSE ≈ 0.28).

In summary, the table reflects a clear trend: as we move from discrete to continuous-action methods (top to bottom), performance improves significantly across all metrics, indicating that DDPG's continuous-action policy learned the task dynamics more efficiently than the discrete baselines.

9 Conclusions

This project, sponsored by Houlihan Lokey, set out to enhance the accuracy and efficiency of convertible bond valuation by improving the Tsiveriotis–Fernandes (TF) model and developing data-driven surrogates. A Crank-Nicolson solver was implemented for the coupled TF partial differential equations and identified systematic pricing biases through comprehensive error analysis. A linear regression correction reduced root-mean-squared error by nearly 90%, and a decision-tree analysis highlighted interest rates, volatility, time-to-maturity, and credit spreads as the dominant drivers of residual error.

To accelerate valuation, a large synthetic dataset was generated reflecting empirical distributions and correlations, then trained supervised learning models—random forests, gradient boosting, and feed-forward neural networks—on millions of TF-priced examples. The best neural network achieved over 75 % of its predictions within 1 % of the benchmark price and delivers microsecond inference times, representing a two-order-of-magnitude speedup over the finite-difference solver.

Reinforcement learning was also explored as a complementary approach for replicating the pricing behavior of the TF model rather than learning explicit exercise strategies, demonstrating proof-of-concept results with Q-learning, DDPG, and actor–critic methods. While these agents learned reasonable exercise policies, further tuning and richer market simulations are required before deployment.

The findings confirm that regression adjustments and supervised-learning surrogates can reproduce TF-level accuracy at greatly reduced computational cost, and that reinforcement learning offers a promising framework for dynamic exercise strategies. Future work will focus on refining PDE boundary conditions, integrating liquidity-adjusted volatility, exploring Physics Implied Neural-Networks (PINNs), improving the accuracy of the best performing Neural Network, and validating the frameworks against historical convertible bond price paths to ensure robustness in live applications.

10 Future Work

Several avenues remain for extending this work. One could first explore the PDE engine itself, in which the near-maturity boundary conditions could be tightened so that the transform-based (TF) solver no longer produces price spikes in the final weeks before expiry. Second, the implied-volatility regressions could be enhanced by incorporating a liquidity overlay. Treating the bid–ask spread as an explanatory variable may help capture the option-smile widening commonly observed under thin trading conditions. Third, once the baseline TF code is more highly calibrated, a larger synthetic dataset can be generated to train machine learning models.

On the machine learning side, a careful analysis should be conducted to determine if the existing 4-million-sample corpus adequately captures the full range of relevant pricing dynamics, or whether additional targeted sampling is required. Furthermore, Physics-informed Neural networks (PINNs), can be explored by embedding the TF equations directly into the loss function. PINNs offer the potential accuracy of finite-difference schemes with orders-of-magnitude faster inference, even in high-dimensional or ill-posed regimes.

The reinforcement learning (RL) framework will also be significantly expanded. Current RL agents have demonstrated strong approximation to TF PDE valuations, achieving over 90% latency reduction while learning robust conversion strategies under synthetic conditions. Future work will explore more advanced actor–critic algorithms, such as A2C or PPO, for improved stability and sample efficiency. There is also scope to investigate batch RL methods, such as Fitted Q-Iteration, which may offer better generalization from offline datasets. Hybrid approaches that combine RL policies with PDE baselines—whether through ensemble learning or reward shaping could be explored to further enhance performance. In addition, incorporating risk-sensitive objectives, such as Conditional Value at Risk (CVaR), will align the agent's behavior more closely with real-world tail-risk management.

This future work will hopefully yield results that improve the valuation of convertible bonds.

11 References

[1] Tsiveriotis, K. and C. Fernandes. (1998, September). Valuing convertible bonds with credit risk. *Journal of Fixed Income*, 8, 95–102.

[2] Goldman Sachs. (1994, November). Valuing convertible bonds with credit risk. *Goldman Sachs Quantitative Strategies Research Notes*.

[3] Hull, J. and White, A. (1995, May). The Impact of Default Risk on the Pricing of Options and Convertible Bonds. *Journal of Derivatives, 3(3), 7–12*.

12 Appendix

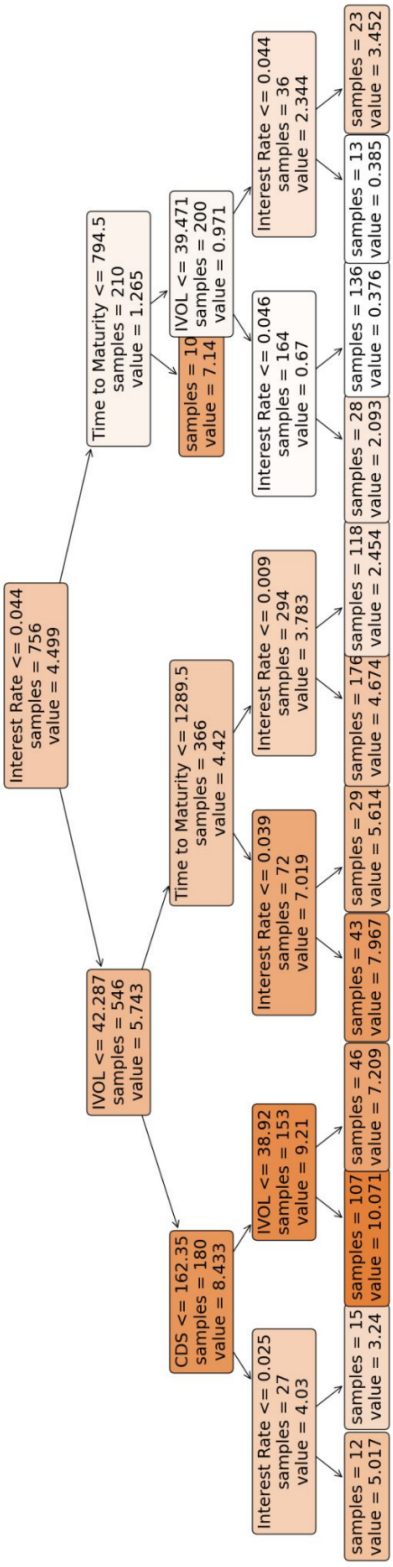


Figure 17: Decision Tree result of implied volatility adjustment

	bond_price	IVOL	CDS	S	BestIVOL	Adjustment
0.0	40.8640	2.62630	2.56	0.011927	0.5	0.2
0.1	83.2460	23.73904	47.50	6.000000	6.7	0.2
0.2	92.2000	30.29092	70.40	11.950000	10.1	0.2
0.3	96.5500	35.92082	83.00	17.110000	13.8	0.2
0.4	99.3988	41.63576	100.50	22.638000	18.8	0.2
0.5	103.0460	47.51105	118.20	29.930000	23.4	0.4
0.6	107.9882	54.05048	138.80	41.611720	28.2	0.6
0.7	115.0761	62.02988	173.60	56.988730	34.2	0.8
0.8	127.0124	73.58248	227.56	87.080000	41.7	1.0
0.9	152.2849	96.43134	387.43	119.926000	54.0	1.2
1.0	1981.7990	2413.26560	1192.60	3901.990000	482.6	2.0

	Interest Rate	RelativeDifference	T	Coupon	Coupon_Freq
0.0	0.001030	-1.263554	56.0	0.125	2.0
0.1	0.003451	-0.178377	579.0	0.375	2.0
0.2	0.005616	-0.078704	834.0	1.000	2.0
0.3	0.009068	-0.025331	1006.0	1.500	2.0
0.4	0.014687	-0.012112	1160.0	2.500	2.0
0.5	0.028487	-0.005562	1322.0	3.250	2.0
0.6	0.037048	-0.000492	1484.0	3.875	2.0
0.7	0.041411	0.004215	1653.0	4.500	2.0
0.8	0.044709	0.011341	1843.0	5.625	2.0
0.9	0.048793	0.033197	2265.0	6.750	2.0
1.0	0.056216	1.000000	10995.0	19.750	4.0

	Conversion_Ratio	Conversion_Price	Dividend_Yield	Coupon_Rate
0.0	0.5315	1.150000	0.0000	0.00125
0.1	5.2809	13.392492	0.0000	0.00375
0.2	8.6073	18.750012	0.0000	0.01000
0.3	15.3227	22.312514	0.0000	0.01250
0.4	24.0763	31.106521	0.0000	0.01500
0.5	31.3475	39.784052	0.0100	0.02375
0.6	40.4040	58.725079	0.0188	0.03250
0.7	51.9224	83.167691	0.0276	0.04000
0.8	69.6767	116.180451	0.0680	0.05375
0.9	116.0227	189.361662	0.1114	0.05750
1.0	4356.8531	1881.467545	0.1817	0.07500

Figure 18: Quantile Analysis Results

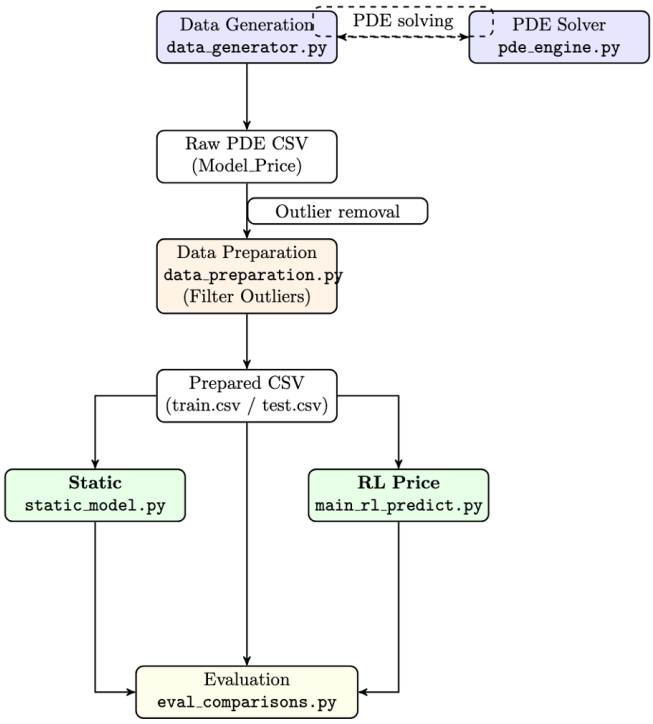


Figure 19: High-level workflow: synthetic-data generation → PDE pricing → RL training (online and static).

12.1 Liquidity analysis for Volatility: Bid-Ask Spread

Bid-ask spreads serve as a crucial proxy for market liquidity and trading friction. In the context of convertible bonds (CBs), a large spread may indicate heightened uncertainty or low daily trading volume, both of which can significantly impact CB pricing, hedging, and overall market quality.

12.1.1 Data Overview and Cleaning

Our dataset contained 85 convertible bonds with around 43.9k observations. Key fields included:

- **PX_LAST** – the last traded price;
- **PX_VOLUME** – daily trading volume;
- **BID** and **ASK** quotes (when available);
- **TTM** (time-to-maturity);
- **Maturity_Date**.

A critical problem was incomplete or missing bid-ask data for approximately 24 bonds. Missing data can complicate any liquidity analysis or subsequent volatility adjustments. We explored two potential strategies:

1. Excluding rows missing bid-ask values,

2. Applying alternative liquidity proxies (e.g., daily volume) for those bonds.

12.1.2 Spread Analysis

We define the spread at time t as:

$$\text{Spread}_t = \text{ASK}_t - \text{BID}_t.$$

We then measured distributional statistics (mean, median, quartiles) and visualized relationships with respect to time-to-maturity (TTM) and volume.

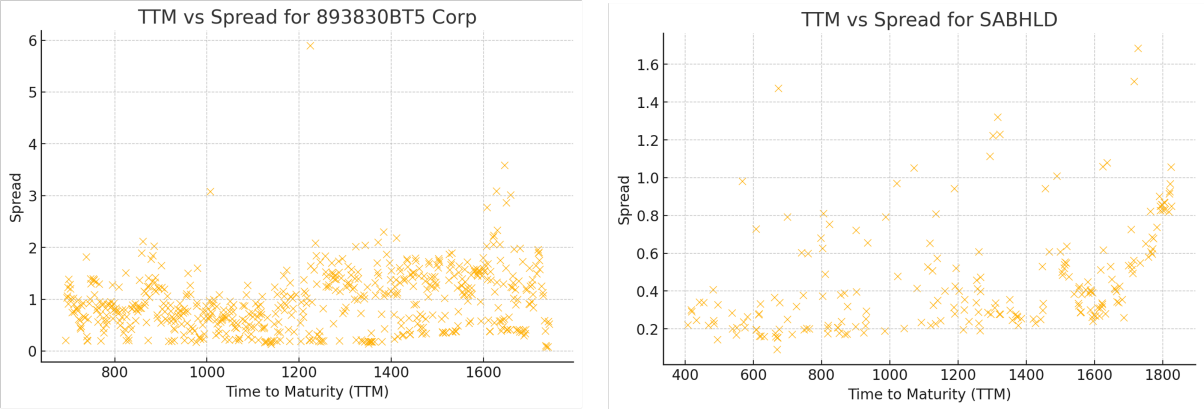


Figure 20: Sample scatter plot of bid-ask spread vs. TTM, highlighting potential correlation with maturity length.

Figure 20 illustrates a scatter of spreads against TTM. From the chart, we observe that while there is no perfectly linear pattern, longer maturities at times exhibit wider spreads. However, short-term notes can also exhibit high spreads if the underlying stock is very volatile or the bond trades infrequently. In addition:

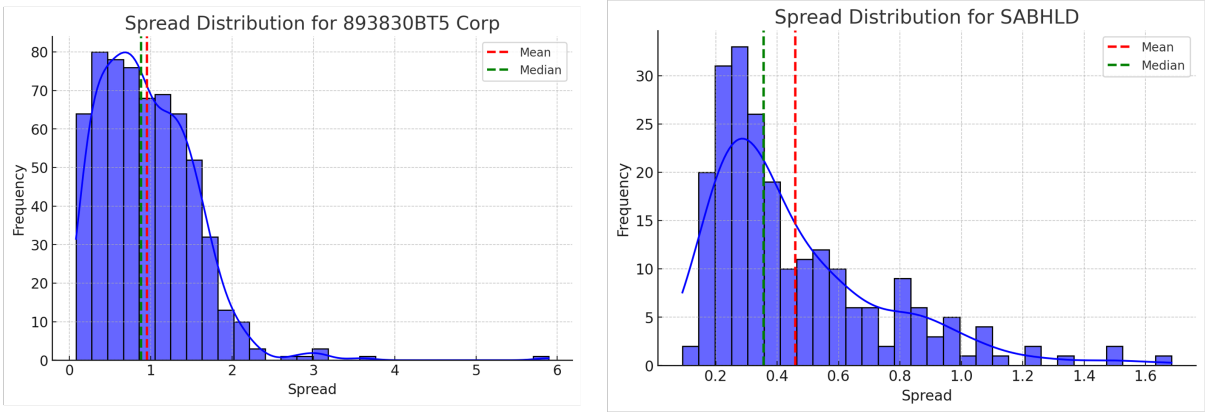


Figure 21: Distribution of bid-ask spreads across the sampled convertible bonds, showing skewness and possible outliers.

In Figure 21, we see the spread distribution with some outliers on the higher end. These outliers often coincide with bonds that have minimal trading volume, reinforcing the interplay between liquidity and observed spreads.

12.1.3 Implications for Pricing

Large bid-ask spreads can distort implied volatility estimates and PDE/binomial-based pricing if one side of the quote is used indiscriminately. Hence, in our PDE or binomial tree calibrations, we typically:

1. Use mid-quotes $\left(\frac{\text{BID}+\text{ASK}}{2}\right)$ to mitigate extremes,
2. Apply upward volatility adjustments when spreads exceed a certain threshold (a proxy for illiquidity),
3. Flag any outlier bonds for deeper qualitative review (e.g., identifying potential corporate events).