# CIR vs Vasicek

Caleb Kilonzi

September 2024

## 1 Report

The provided code is a Python script that simulates and compares the behavior of two widely used interest rate models: the Cox-Ingersoll-Ross (CIR) model and the Vasicek model. These models are commonly employed in finance to simulate interest rate dynamics and volatility over time. The script utilizes numpy for numerical computations and matplotlib to plot the final results.

The code begins by importing two essential libraries, numpy and matplotlib.pyplot. numpy is used to perform array operations and to generate random values, while matplotlib.pyplot is used to create and display the histogram of the results. After the import statements, several parameters are defined, which will serve as the input for both models. These include kappa (the rate at which the process reverts to its long-term mean), v0 (the initial value of the process), v_bar (the long-term mean value), sigma (the volatility of the process), T (the total time period for the simulation), N (the number of steps or intervals), and M (the number of paths generated in the Monte Carlo simulation). The parameter dt is also calculated, representing the time step for each interval, computed as $T/N$.

Next, the script defines two functions to generate paths for the CIR and Vasicek models. The function generate_CIR_paths simulates the CIR model by solving its corresponding stochastic differential equation (SDE). The CIR model, known for enforcing non-negativity in interest rates or volatility, is useful when simulating processes that cannot go below zero, such as interest rates. The model is governed by the equation:

$$dv_t = \kappa(\bar{v} - v_t)dt + \sigma\sqrt{v_t}dW_t$$

where $v_t$ is the value of the process at time $t$, and $dW_t$ is the increment of a Wiener process (Brownian motion). The code simulates this process over multiple paths (M paths) and time steps (N steps), updating the process value at each step based on the equation. It uses random normal variables Z to generate the Brownian motion increments (dW) and ensures that the values of the process remain non-negative by applying a np.maximum function, which prevents negative values from occurring.

Following this, the script defines another function, generate_Vasicek_paths, to simulate the Vasicek model, another mean-reverting process often used in finance to model interest rates.

Unlike the CIR model, the Vasicek model allows for negative values, which can be problematic for interest rate modeling but is useful in certain contexts. The governing SDE for the Vasicek model is:

$$dv_t = \kappa(\bar{v} - v_t)dt + \sigma dW_t$$

The key difference between the CIR and Vasicek models is that the Vasicek model does not include the square root term, and thus, the process does not inherently prevent negative values. Similar to the CIR model, the function generates multiple paths by iterating over time steps and updating the process values according to the SDE.

After defining the two functions, the script proceeds to generate the paths for both models by calling generate_CIR_paths and generate_Vasicek_paths with the parameters specified earlier. These functions return matrices, where each row represents a simulated path and each column corresponds to a time step. The script then extracts the final values (i.e., the values at the last time step) from both sets of paths. These final values are of particular interest because they represent the outcome of each path after the entire simulation period.

The final part of the code focuses on visualizing the results. It uses matplotlib to plot histograms of the final values from both the CIR and Vasicek simulations. These histograms represent the probability density functions (PDFs) of the final values, allowing the user to compare the behavior of the two models visually. The histograms are plotted with a transparency (alpha=0.6) to allow for easy comparison between the two distributions. The plot is titled "PDF of Final Values of CIR and Vasicek Models" and includes a legend to distinguish between the two models. Finally, the plot is saved as a PDF file (CIR_vs_Vasicek_PDF.pdf), and the show() function is called to display the plot.

The results of the simulation, as visualized in the provided PDF, offer several important insights. The histogram for the CIR model exhibits a right-skewed distribution, with a noticeable concentration of values near zero. This skewness is expected due to the non-negativity constraint inherent in the CIR model. Since the model forces the values to remain positive, the distribution tends to cluster around lower values, although there is a tail extending to the right, representing the possibility of higher values occurring over time. This behavior reflects the mean-reverting nature of the CIR model combined with its prevention of negative rates, making it suitable for modeling processes like interest rates where negative values are not realistic.

In contrast, the histogram for the Vasicek model shows a more symmetric distribution centered around zero. This normal-like distribution results from the model's allowance for negative values, which can occur when simulating processes such as interest rates. The Vasicek model's flexibility to generate negative outcomes contrasts with the CIR model's restriction to non-negative values. This makes the Vasicek model less suitable for modeling interest rates in certain contexts, though it might be preferred when modeling other financial quantities where negative values are plausible.