Alternative Methods for Simulating Square Root Processes

Louis Scott

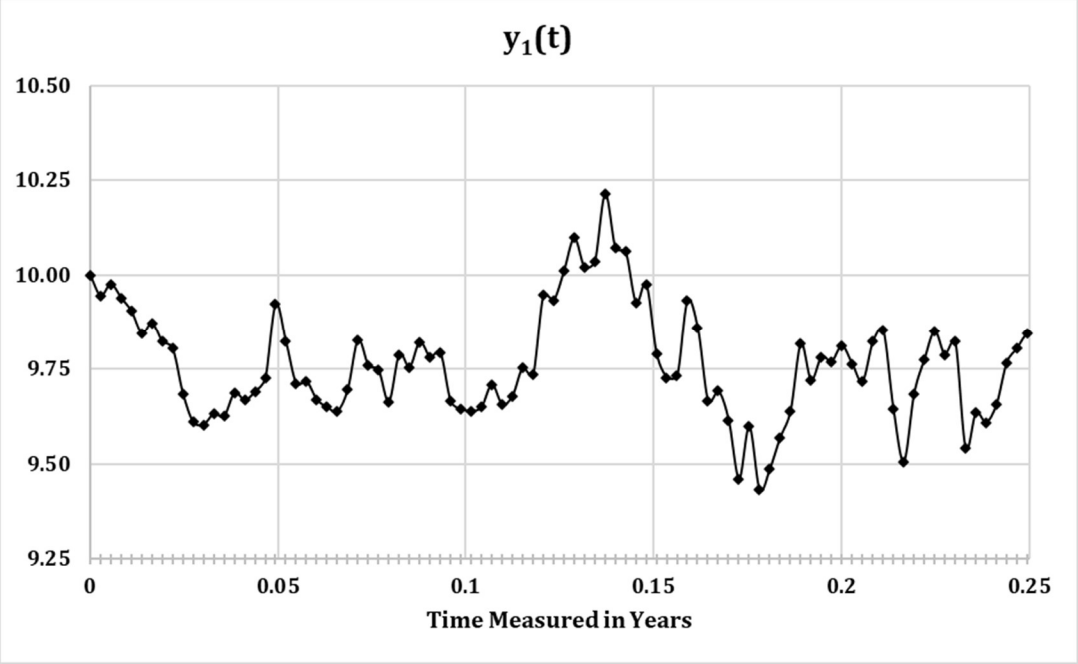
November 2024

Square root processes are used in quantitative finance to model state variables that are nonnegative. Applications include models for stochastic volatility and stochastic credit spreads, as well as the Cox-Ingersoll-Ross (1985) model which uses the square root process to model strictly nonnegative interest rates. This process produces models that are analytically tractable and useful for modeling financial time series. In derivative pricing, there are many cases which require Monte Carlo simulation of stochastic processes, including the square root process. The purpose of this paper is to reexamine methods for simulating square root processes over small time periods in dynamic financial models for applications with parallel processing.

Diffusion processes in financial models have the following form:

$$\boldsymbol{dy} = \boldsymbol{\mu}(\boldsymbol{y}, t)\, dt + \boldsymbol{\sigma}(\boldsymbol{y}, t)\, \boldsymbol{dz}\,,$$

where the bold variables represent vectors and $\boldsymbol{dz}$ is the change for a vector of Brownian motion processes, each with mean zero and variance dt. Simulations of key financial state variables (asset prices, interest rates, variance rates, …) typically use daily time steps and extend for months or many years, depending on the application and the time horizon. The following graph depicts an example of a financial state variable simulated over a 3-month time-period with daily time steps.



Simulations for the Brownian motions rely on random number generators to produce sequences of normally distributed random numbers. The structures for the instantaneous means and standard deviations, $\boldsymbol{\mu}(\boldsymbol{y}, t)$ and $\boldsymbol{\sigma}(\boldsymbol{y}, t)$, include the model parameters. In derivative pricing models, simulations are used to approximate expected values for valuation, as well as partial derivatives which are the deltas used for hedging. The simulations are also used to calibrate the model parameters to fit model valuations to market prices. The deltas for hedging and the model calibration require repeated simulations of the same random numbers for $\boldsymbol{dz}$ to assess the impacts of either changes in initial state variables for hedging or different parameter values for calibration. If the same random number sequence is used to evaluate partial derivatives for hedging, the simulation error for the deltas will be significantly reduced. The solutions for model calibration will be slightly different for each sequence of random numbers, so that a user must control the random number generation to repeat the same set of random simulations for $\boldsymbol{dz}$. For these reasons, pseudo number generators (RNG) are typically used, and the initial seeds for the RNGs are controlled to ensure that the simulations for $\boldsymbol{dz}$ are repeated.

Computing technology has improved dramatically in recent years so that financial models requiring Monte Carlo simulation can run very quickly with parallel processing using either multiple threads on CPU's, GPU's, or special purpose processing chips. In the case of Monte Carlo simulation for derivative pricing, fast methods are necessary to produce valuations and hedges in real time as markets change. Parallel processing introduces additional requirements for the simulation of random numbers, as most RNGs are sequential. The pseudo RNG's have initial seeds or states that can be used to control the sequences of random numbers. For some of these RNG's, there are quick methods, step-ahead techniques, for calculating the initial seeds necessary for subsequent simulations across multiple processors, which will run in parallel.[1] These step-ahead techniques make it possible to control the initial seeds and the random number simulations necessary for testing, calculating deltas, and parameter calibration. Parameter calibration has typically been a challenge for models that must be solved by simulation. In the past, the parameter calibration was performed with approximation models, as Monte Carlo methods were too slow. This is no longer the case as massive parallel processing with GPU's facilitates fast simulation so that one can now perform real time pricing and parameter calibration with simulation models. The set of parameters that optimizes an objective function for the fit of model prices to market prices changes if the simulated random numbers change. For this reason, it is important to run an optimization algorithm with a fixed set of random numbers. If one is not able to control the random simulations, the optimization algorithm will be chasing optimal parameter sets that vary. This is the "scrambling" effect mentioned in Andersen, Jäckel, and Kahl (2010). If one uses the same starting seeds for a uniform RNG, then the same random numbers can be re-

[1] This is the case for the pseudo RNG's covered in L'Ecuyer (1999), L'Ecuyer and Simard (2007), and Salmon, Moraes, Dror, and D. E. Shaw (2011). Pseudo RNG's from these references are used for parallel processing in the statistical libraries available in python and Cuda C.

produced as model parameters are adjusted. Likewise, one can control a sequence of normal or exponential random numbers by simulating uniform random numbers and using the inverse of the cumulative distribution function (CDF). To control simulations across parallel processing, it is simply a matter of counting the number of simulations on each simulation path and applying the appropriate step-ahead method to calculate the initial seeds for the RNG on each processor. In this manner, one can use parallel processing to reproduce a set of simulations for uniform, exponential, and normal random numbers for hedging and parameter calibration.

Section 1 reviews the square root process and the simulation methods based on the non-central chi squared distribution, which is the probability distribution for the process over discrete time intervals. Simulation methods for the non-central chi squared distribution are relatively slow and are not well suited for parameter calibration and parallel processing. As discussed in section 1, non-central chi squared simulation requires acceptance/rejection methods so that it is difficult to predict the number of simulations for each time step and control the number of simulations on each simulation path. Section 2 covers alternative approximations methods and develops an extension of existing methods. The alternative simulation methods require only one simulation per time step and facilitate both parameter calibration and parallel processing. Section 3 provides tests of the alternative simulation methods, including applications with parallel processing on a GPU. The test results indicate that the approximation methods generally converge and significantly reduce computing time when parallel processing is applied.

I. Simulating Square Root Processes

The square root process is a diffusion process with the following stochastic differential equation.

$$dx = (\kappa\theta - (\kappa + \lambda)x)\, dt + \sigma\sqrt{x}\, dz\,, \tag{1}$$

where dz is the change in a Brownian motion. The distribution of $x(t)$, conditional on $x(0)$ for $t > 0$, is non-central chi squared. The parameters κ, θ, and σ are greater than zero, and λ is a risk premium parameter that can be positive or negative. θ is the long run mean and κ determines the rate of mean reversion for the "real world" process ($\lambda = 0$). $\kappa + \lambda$ determines the rate of mean reversion under the risk neutral measure. If $\kappa + \lambda \leq 0$, the risk-neutral process for x is not a stationary time series, but one can evaluate expectations for asset pricing using the distribution conditional on a starting value. This process is used in quantitative finance for random variables that are restricted to be nonnegative. Various applications include Heston's (1993) stochastic volatility model, the Cox-Ingersoll-Ross (CIR, 1985) model for interest rates, and the stochastic volatility LIBOR market models developed in Andersen and Brotherton-Ratcliffe (2005), Wu, and Zhang (2006), and Kiesel and Lutz (2011). Duffie, Pan, and Singleton (2000) developed an exponential affine model for option prices in which an independent jump process is added to the square root process for the stochastic volatility.

One method for simulating x is to use the exact distribution and simulate a non-central chi squared variate over discrete time intervals. The variable $c\,x(t)$, conditional on $x(0)$, is distributed as a non-central chi squared with ν degrees of freedom and non-centrality parameter λ_{nc} defined as follows:

$$\nu = \frac{4\kappa\theta}{\sigma^2}\,, \qquad \lambda_{\text{nc}} = cx(0)e^{-(\kappa+\lambda)t}\,, \qquad \text{and} \quad c = \frac{4(\kappa+\lambda)}{\sigma^2(1-e^{-(\kappa+\lambda)t})}\,.$$

There are several methods available for simulating non-central chi squared variates. The distribution is a mixture of the chi squared distribution with the degrees of freedom generated randomly by a Poisson distribution, so that one method is to simulate a Poisson variate with parameter $\frac{1}{2}\lambda_{\text{nc}}$ followed by a simulation of a chi squared with degrees of freedom equal to ν plus 2 times the Poisson simulation. Another method covered in Johnson & Kotz (1995, Ch. 29) and Glasserman (2003, Ch. 3), is to simulate a normal variate and a chi squared variate as follows, with ν degrees of freedom and non-centrality parameter λ_{nc}:

$$\chi'^2\,(v, \lambda_{\text{nc}}) = \left(Z + \sqrt{\lambda_{\text{nc}}}\right)^2 + \chi^2(v-1)$$

where Z is a standard normal random variate and $\chi^2(v-1)$ is a chi squared with $v-1$ degrees of freedom. $\chi'^2\,(v, \lambda_{\text{nc}})$ is used to indicate a non-central chi squared distribution. This latter method requires $v \geq 1$. The first method using a Poisson simulation and a chi squared simulation can be used for any $v > 0$. The exact method requires either a Poisson simulation plus a chi squared simulation or a normal simulation plus a chi squared simulation. Poisson simulations and normal simulations can be calculated very quickly, but chi squared simulation is more complex and typically requires acceptance-rejection methods. These methods are exact, but require significantly more computing time, relative to the simulation of random variables with normal or exponential distributions, and the number of random number simulations varies at each time step. The exact simulation applied to the square root process is appropriate for any chosen time interval, but it does not take advantage of the property of that a diffusion process is approximately normal over small time intervals. In addition, exact simulation of the non-central chi squared distribution uses acceptance/rejection methods so that the number of simulations for each time step vary and one cannot predict or control the number of simulations required for each simulation path. To perform parameter calibration, testing, real time pricing, and hedging with the simulation of square root processes, it is necessary to have a fast method which includes control over the initial seeds and the number of simulations for a selected RNG.

2. Alternative Approximation Methods

Stochastic differential equations are approximated by simulating over small discrete time intervals. Financial models for equity prices and FX rates, for example, typically use simulations over daily time intervals. The simulation of an exact non-central chi squared

over a small time interval does not take advantage of the property that a diffusion process is a function of a Brownian motion, which has a normal distribution. The papers by Kahl and Jäckel (2006), Andersen (2008), and Andersen, Jäckel, and Kahl (2010) develop several alternative methods for the simulation of square root processes. Two of their methods are particularly useful for simulating square root processes: Andersen's (2008) quadratic-exponential method (QE) and the implicit Milstein scheme in equation (11) of Andersen, Jäckel, and Kahl (2010), which was originally developed in Kahl and Jäckel, (2006). The QE method uses either a quadratic function of a normal simulation or a simulation based on a mixture of an exponential with a distribution concentrated at zero. The mixture of the exponential simulation and the simulation of zeros is used to capture the behavior of the non-central chi squared distribution for values close to zero when the degrees of freedom parameter is small. The QE method is a moment matching method in which the model parameters are set to match the mean and variance of the diffusion over a small time interval. The implicit Milstein scheme in is useful when the degrees of freedom parameter is greater than or equal to one. It should be noted that the implicit Milstein scheme is similar, but not identical, to the quadratic simulation in Andersen's QE method.

An alternative approach is to transform the diffusion equation and use a solution to a differential equation that has a constant variance for the diffusion term. For the square root diffusion, one can apply the following transformation to x.

$$y = \sqrt{x}$$

An application of Ito's lemma produces the following stochastic differential equation for y.

$$dy = \frac{1}{2}\left(\frac{\left(\kappa\theta - \sigma^2/4\right)}{y} - (\kappa+\lambda)y\right)dt + \frac{\sigma}{2}\,dz_t$$

The second term is a normal random variable, and an exact solution for y must satisfy this stochastic differential equation. If $4\kappa\theta = \sigma^2$, the differential equation can be solved as follows:

$$y(t) = e^{-\frac{1}{2}(\kappa+\lambda)t}y(0) + \frac{\sigma}{2}\int_0^t e^{-\frac{1}{2}(\kappa+\lambda)(t-s)}dz_s\,.$$

The solution for $4\kappa\theta \neq \sigma^2$, is more complex and does not lead to a tractable solution for simulating y or ultimately x. An approximate solution can be found by working with the solution to the following ODE:

$$dy = \frac{1}{2}\left(\frac{\left(\kappa\theta - \sigma^2/4\right)}{y} - (\kappa+\lambda)y\right)dt\,,$$

which is

$$y(t) = \sqrt{e^{-(\kappa+\lambda)t}y^2(0) + \left(\kappa\theta - {}^{\sigma^2}/_4\right)\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)}\,.$$

This leads to the following proposed approximation:

$$y(t) = \sqrt{e^{-(\kappa+\lambda)t}y^2(0) + \left(\kappa\theta - {}^{\sigma^2}/_4\right)\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)} + \frac{\sigma}{2}\int_0^t e^{-\frac{1}{2}(\kappa+\lambda)(t-s)}dz_s\,.$$

$y(0) = \sqrt{x(0)}$ and $x(t) = y^2(t)$ so that the solution can be expressed in terms of x as follows:

$$x(t) = \left(\sqrt{e^{-(\kappa+\lambda)t}x(0) + \left(\kappa\theta - {}^{\sigma^2}/_4\right)\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)} + \frac{\sigma}{2}\int_0^t e^{-\frac{1}{2}(\kappa+\lambda)(t-s)}dz_s\right)^2. \quad (2)$$

It should be noted that this simulation method is a quadratic function of a normal simulation, which is similar to the implicit Milstein scheme and the quadratic part of the QE method described above. The integral of the diffusion process is simulated by simulating a standard normal random variate and multiplying by the square root of the variance,

$$\text{Var}\left(\int_0^t e^{-\frac{1}{2}(\kappa+\lambda)(t-s)}dz_s\right) = \left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right).$$

The mean and variance of $\Delta x \equiv x(t) - x(0)$ for the square root process, for a given time interval t, are as follows:

$$E(\Delta x) = \left(e^{-(\kappa+\lambda)t} - 1\right)x(0) + \kappa\theta\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right) \quad (3)$$

$$\text{Var}\,(\Delta x) = \sigma^2\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)\left(e^{-(\kappa+\lambda)t}x(0) + \tfrac{1}{2}\kappa\theta\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)\right). \quad (4)$$

These two moments follow from the non-central chi squared distribution for $x(t)$, conditional on $x(0)$. The evaluation of the mean and variance for the approximation model follows from an analysis of the following terms.

$$x(t) = e^{-(\kappa+\lambda)t}x(0) + \kappa\theta\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)$$

$$+ \sqrt{e^{-(\kappa+\lambda)t}x(0) + \left(\kappa\theta - {}^{\sigma^2}/_4\right)\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)}\;\sigma\int_0^t e^{-\frac{1}{2}(\kappa+\lambda)(t-s)}dz_s$$

$$- \frac{\sigma^2}{4}\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right) + \left(\frac{\sigma}{2}\int_0^t e^{-\frac{1}{2}(\kappa+\lambda)(t-s)}dz_s\right)^2$$

The approximation model for the simulation of Δx has a mean that matches the mean for the square root process. The variance for Δx in the approximation model is

$$\begin{aligned}\text{Var}\,(\Delta x) &= E\left(\sqrt{e^{-(\kappa+\lambda)t}x(0) + \left(\kappa\theta - \sigma^2/4\right)\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)}\ \int_0^t \sigma e^{-\frac{1}{2}(\kappa+\lambda)(t-s)}dz_s\right.\\ &\qquad \left. - \frac{\sigma^2}{4}\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right) + \left(\frac{\sigma}{2}\int_0^t e^{-\frac{1}{2}(\kappa+\lambda)(t-s)}dz_s\right)^2\right)^2\\ &= \sigma^2\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)\left(e^{-(\kappa+\lambda)t}x(0) + \left(\kappa\theta - \frac{\sigma^2}{4}\right)\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)\right)\\ &\qquad +2\left(\frac{\sigma^2}{4}\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)\right)^2\\ &= \sigma^2 x(0)t + \mathrm{o}(t)\end{aligned}$$

The variance for the approximation model in (2) does not match the variance for the non-central chi squared distribution, but it does converge to the variance for the square root process over small time intervals as t approaches 0. Variance is an important characteristic for a distribution, and it is desirable to have approximation methods that match both the mean and variance over small time steps. The variance bias is easy to compute, and the approximation model can be modified to match the variance, by applying the following additional calculations: subtract the mean from $x(t)$, rescale this residual so that the variance matches Var (Δx), and add back the mean. The rescaling factor is calculated as follows.

$$\sqrt{\frac{e^{-(\kappa+\lambda)t}x(0) + \frac{1}{2}\kappa\theta\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)}{e^{-(\kappa+\lambda)t}x(0) + \left(\kappa\theta - \sigma^2/8\right)\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)}}$$

If $4\kappa\theta \geq \sigma^2$, the scaling factor is less than 1 and the resulting calculation for $x(t)$ is guaranteed to be nonnegative. If $4\kappa\theta < \sigma^2$, the scaling factor is greater than 1, and the adjustment cannot be applied if the resulting value for $x(t)$ is negative. For the case $4\kappa\theta = \sigma^2$, the approximation model in (2) is an exact simulation of the non-central chi squared distribution, and the rescaling factor is 1.

If $4\kappa\theta < \sigma^2$, the term inside the square root in (2) can become negative for small values of $x(0)$. In addition, if the square root term is nonnegative, the rescaling to match the variance can produce a negative value for x. In these cases, an alternative simulation method is required. For cases where $4\kappa\theta < \sigma^2$, one can apply Andersen's QE method, but this methodology tends to produce many simulations at exactly 0. In cases where $2\kappa\theta < \sigma^2$, the square root process can touch zero which serves as a reflecting barrier (See Feller (1950)). In these cases, the CDF is 0 for x=0, or $F(0) = 0$, with the probability density function going to infinity as x approaches 0. The slope of $F(x)$ at $x = 0$ is infinite. For cases where the quadratic method cannot be applied, one must develop an approximation method that approximates the properties of the distribution for the square root process. The remainder of this section is devoted to the development of several alternative methodologies, which can be applied with the quadratic method in (2) with rescaling to match the variance or Andersen's quadratic method to directly match the mean and variance.

The alternative approximation methodologies should match the mean and the variance for the square root process and provide approximations for the relevant non-central chi squared distribution. Several approximation methods have been developed and examined, and five of these methods are included here. The alternative methods can use either the quadratic method in (2), with rescaling to match the exact variance for the square root process, or the moment matching quadratic method in Andersen's QE method. The alternative methods differ in the treatment for the case where the quadratic method cannot be used. The first alternative methodology is a mixture of the non-central chi squared distribution with 1 degree of freedom and a beta distribution as follows.

$$F(cx) = p\ (\min\,[cx, 1])^{\nu^*/2} + (1-p)\chi^{2\prime}\left(1, \lambda_{nc}\,, {}^{cx}/_{\beta}\right) \quad \text{for } x \geq 0$$

$\chi^{2\prime}(1, \lambda_{nc}\,, cx/\beta))$ is the CDF for the non-central chi squared distribution with one degree of freedom, non-centrality parameter λ_{nc}, and input argument cx/β. The beta distribution in $(\min\,[cx, 1])^{\nu^*/2}$ is close to a chi squared with degrees for freedom less than 1 as it has the property that the density function goes to infinity as x gets close to zero. The beta distribution is also used in the exact simulation of a chi squared when the degrees of freedom is small. The non-central chi squared with one degree of freedom is used because it is easy to simulate (one normal simulation) and it provides a good approximation for values of ν below, but near one. The beta parameter ν^* is scaled down from ν as follows: $\nu^* = \nu$ / 2.0. Simulation is performed by using the inverse function method: simulate a uniform random variable for $F(cx)$ and invert to get the simulated value for x. The parameters p and β are set to match the mean and variance for the square root process. See the Appendix, section A, for the calculations for p and β and the simulation using the inverse CDF method. The proposed simulation methodology uses either a quadratic function of a normal simulation or a mixture of a beta and a non-central chi squared with one degree of freedom. Hence, the name Quadratic–Beta–Non-Central Chi Squared or QBNC1 is applied. In summary for cases where $\nu < 1$, the QBNC1 methodology uses the normal simulation in the quadratic method when the moment matching conditions are satisfied. If these conditions for the quadratic

method are not satisfied, switch to the mixed Beta Non-Central Chi Squared simulation in equation (5), with p and β as specified in the Appendix A.

The second alternative method uses an approximation for the CDF of the Non-Central Chi Squared when the moment matching conditions for the quadratic method are not satisfied. The CDF approximation uses a beta distribution and 2 exponential distributions, hence the name Quadratic–Beta–Double Exponential or QB2Exp is applied. Observe that the Non-Central Chi Squared distribution is a mixture of the Chi Squared distribution with the following CDF and PDF.

$$F(cx; \nu, \lambda_{nc}) = e^{-\frac{1}{2}\lambda_{nc}} \sum_{j=0}^{\infty} \frac{\left(\frac{1}{2}\lambda_{nc}\right)^j}{j!} \chi^2(cx;\ \nu + 2j)$$

$$f(cx; \nu, \lambda_{nc}) = e^{-\frac{1}{2}\lambda_{nc}} \sum_{j=0}^{\infty} \frac{\left(\frac{1}{2}\lambda_{nc}\right)^j}{j!} f_{\chi^2}(cx;\ \nu + 2j)$$

$$f_{\chi^2}(cx;\ \nu + 2j) = \frac{c\ (cx)^{\frac{\nu}{2}+j-1}\ e^{-\frac{1}{2}cx}}{2^{\frac{\nu}{2}+j}\ \Gamma\left(\frac{\nu}{2}+j\right)}$$

When the quadratic method cannot be applied, the non-centrality parameter λ_{nc} is small, and the PDF can be approximated with the first term, j = 0. When cx is sufficiently small, $\exp(-0.5cx)$ is approximately equal to 1, and the following approximation can be used for the CDF.

$$F(cx; \nu, \lambda_{nc}) \approx F^*(x; \nu, \lambda_{nc}) = \frac{e^{-\frac{1}{2}\lambda_{nc}}}{2^{\frac{\nu}{2}}\ \Gamma\left(\frac{\nu}{2}\right)} \frac{2}{\nu}\ (cx)^{\frac{\nu}{2}}$$

For $x \leq x_1$:

$$F^*(x) = \frac{e^{-\frac{1}{2}\lambda_{nc}}}{2^{\frac{\nu}{2}}\ \Gamma\left(\frac{\nu}{2}\right)} \frac{2}{\nu}\ (cx)^{\frac{\nu}{2}} = \frac{2}{\nu}\ A^*\ (cx)^{\frac{\nu}{2}}$$

$$f^*(x) = c\ A^*\ (cx)^{\frac{\nu}{2}-1}$$

with

$$A^* = \frac{e^{-\frac{1}{2}\lambda_{nc}}}{2^{\frac{\nu}{2}}\ \Gamma\left(\frac{\nu}{2}\right)}$$

The CDF for $x > x_1$ is approximated with 2 exponential distributions as follows.

For $x_1 < x \leq x_2$

$$F^*(x) = (1 - F^*(x_1)\left(1 - e^{-\beta_1(x-x_1)}\right)$$

$$f^*(x) = (1 - F^*(x_1)\,\beta_1\, e^{-\beta_1(x-x_1)}$$

For $x > x_2$

$$F^*(x) = (1 - F^*(x_2)\left(1 - e^{-\beta_2(x-x_2)}\right)$$

$$f^*(x) = (1 - F^*(x_2)\,\beta_2\, e^{-\beta_2(x-x_2)}$$

β_1 and β_2 are set to match the mean and variance of the approximation model to the mean and variance for the square root process. x_1 is set so that the second term in the expansion of $\exp(-0.5cx)$ is equal to -0.1. x_2 is equal to x_1 plus the mean for x. The detailed calculations for the mean of x, the variance of the change in x, β_1, and β_2 are in Appendix B. The solutions for β_1 and β_2 that match the mean and variance require an iterative search on the value for β_1. This Beta Double Exponential method is based on a piecewise approximation of the Non-Central Chi Squared CDF when the moment matching conditions for the quadratic method are not satisfied. The iterative search procedure and the inversion from the approximation CDF's can be calculated very quickly to produce model simulations that approximate the square root process.

The QBNC1 and the QB2Exp methods are recommended for simulation of the square root process when the degrees of freedom parameter, ν, is less than 1, and both methods provide good approximations to the Non-Central Chi Squared distribution. Additional alternative methods have been investigated but are not included in this version of the paper. Additional tests have been run using both the exact simulation of the Non-Central Chi Squared distribution described in section 1 and simulation based on the inversion of the CDF for the Non-Central Chi Squared distribution in all cases. The exact methods are much slower if simulated over small time steps.

3. Tests of the Simulation Methodologies

A variety of comparisons and tests are presented in this section beginning with plots of the CDFs for the alternative approximation methods with the exact distribution for a non-central chi squared with degrees of freedom equal to 0.08. This is the case covered in Figures 1 and 2 in Andersen (2008), and in Figure 1 in Andersen, Jäckel, and Kahl (2010). Figure 1 here contains plots of the distribution functions with the model parameters set as follows: $\kappa = 0.5, \theta = 0.04, \sigma = 1.0\ \lambda = 0$, $t = 0.1$, and $x(0) = 0.01$. In this case, an alternative distribution is used to approximate the non-central chi squared because the quadratic approximation cannot be used. The distribution functions for the approximation methods QB2Exp and QE are close to the exact distribution functions, but there are some differences, notably the concentration of the QE distribution function at $x = 0$. The approximation methods

improve as the size of the time interval is reduced. As previously noted, derivative pricing models typically use daily time steps so that the size of the time interval is either $t = {}^{1}/_{365}$ for a calendar day, or with $t = {}^{1}/_{252}$ for a trading day. The graphs in Figure 2 contain plots of the distribution functions for daily time steps with $t = {}^{1}/_{365}$, and the model parameters used in Figure 1. The distribution functions for both approximation methods are much closer to the exact distribution when the time interval is reduced to one day, with one exception: the case in Panel B in which the QE method produces a high probability for $x = 0$. This is a characteristic of the QE approximation method.

Figure 1. Cumulative Distribution Functions for Approximation Methods

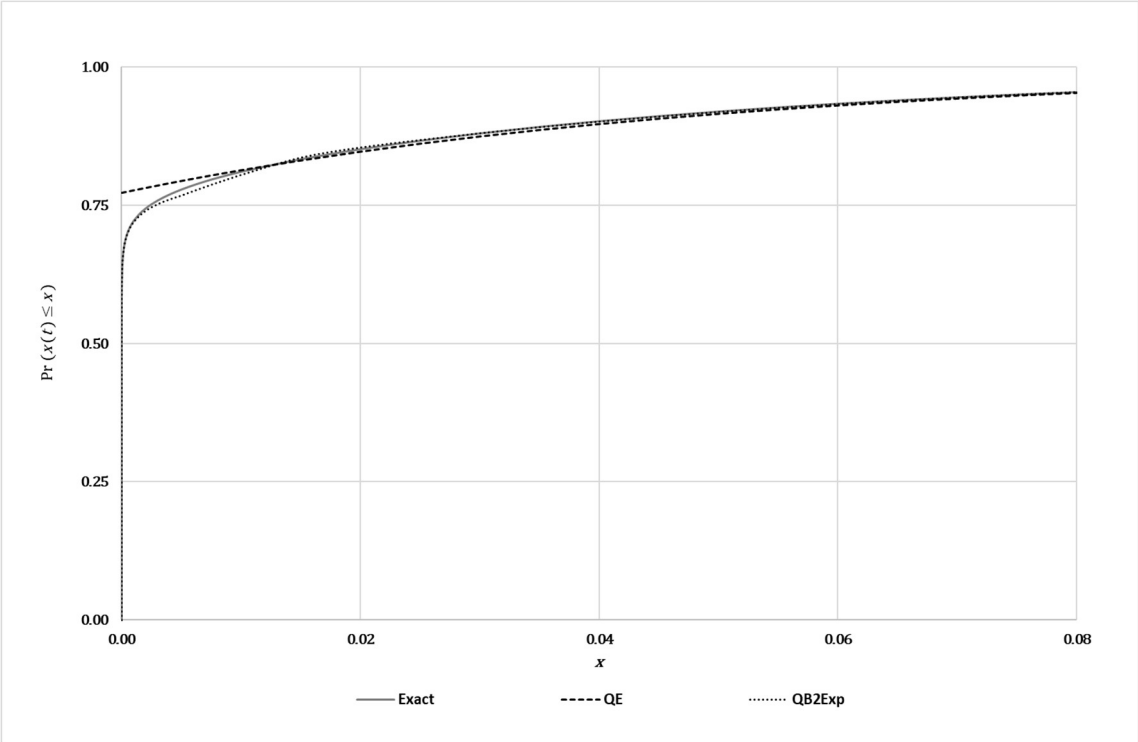
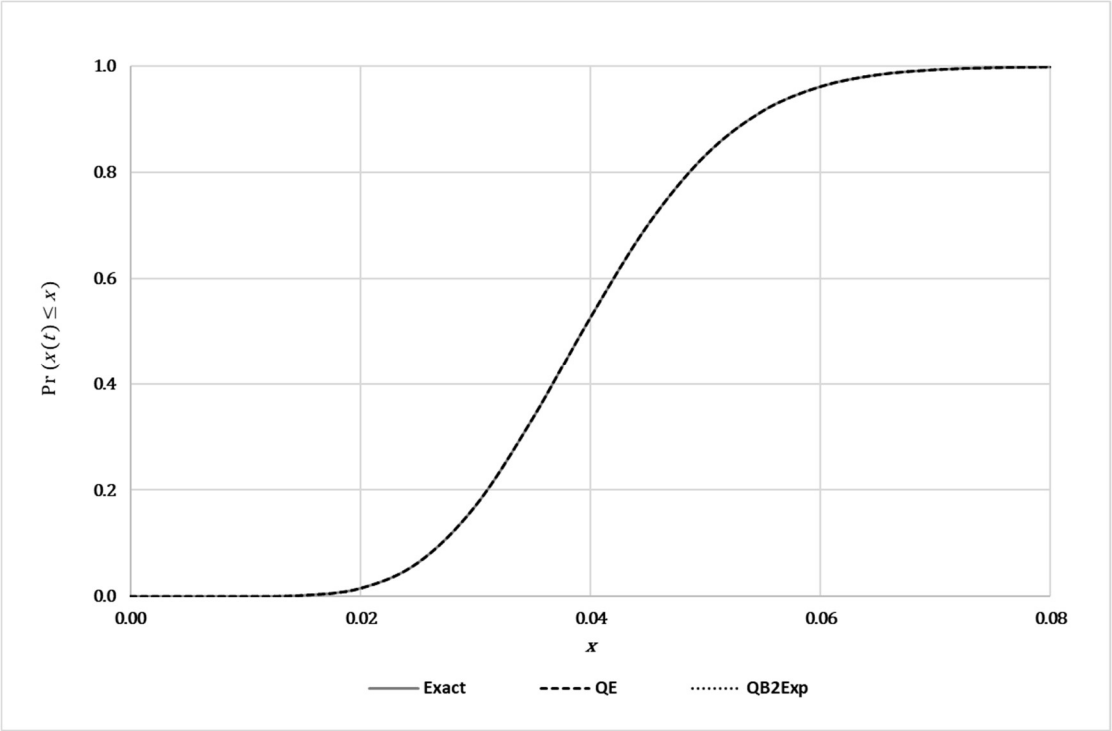
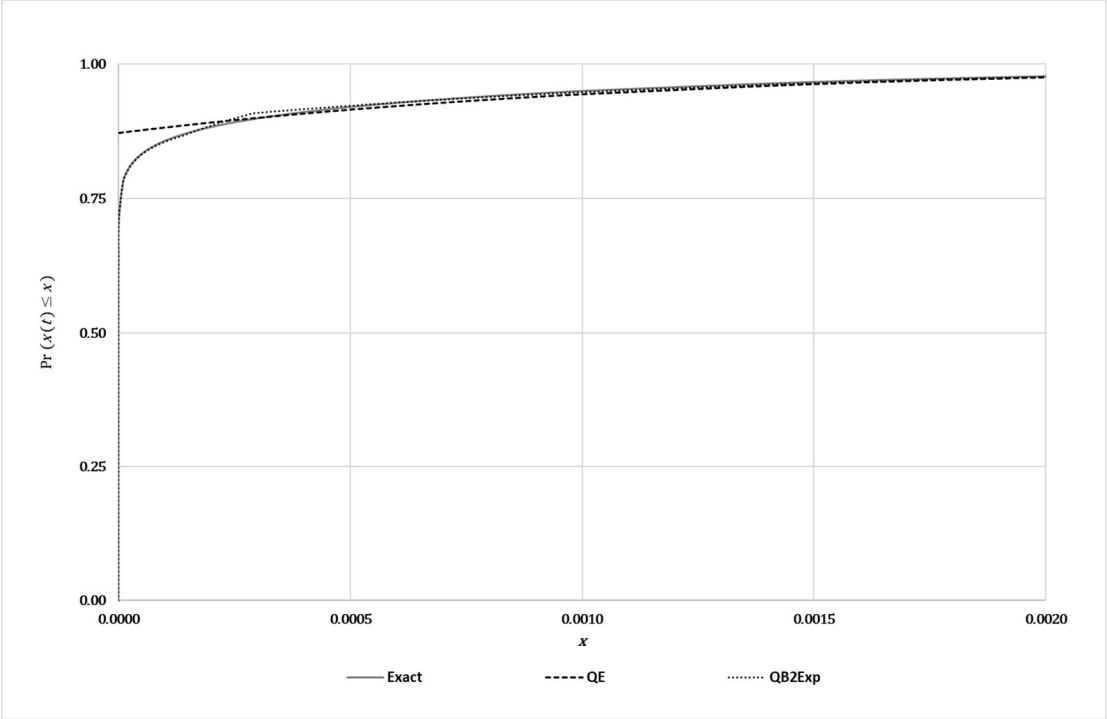


Figure 2. Cumulative Distribution Functions for a Daily Time Interval

Panel A. $x(0) = 0.04$



Panel B. $x(0) = 0.0001$



The next set of tests are tests of the simulated distributions generated by the three approximation methods, as approximations for the non-central chi squared distribution. The Euler methodology is included in these tests as a "straw man" for comparison purposes. The Euler method is simple and easy to implement, and in the cases tested, it requires less computing time. The following equation is used for the Euler methodology:

$$x(t) = x(0) + \left(e^{-(\kappa+\lambda)\Delta t} - 1\right)x(0) + \kappa\theta\left(\frac{1 - e^{-(\kappa+\lambda)\Delta t}}{\kappa + \lambda}\right) + \sigma\sqrt{x(0)}\,\Delta z$$

where Δz is a normal random variable with mean zero and variance Δt, and $x(t)$ is set to 0 if a negative value is simulated. When the degrees of freedom of parameter are less than 2, the Euler method produces negative values. When the degrees of freedom are less than 1, the method produces simulations with significant biases.

The simulation methods are run with the following model parameters for daily time steps: $\kappa = 0.25,\ \theta = 0.04, \lambda = -.125$, and $t = \frac{1}{365} = 0.00273973$, with different values for $x(0)$: 0.01 and 0.04. The values for σ are varied to produce a range of values for the degrees of freedom, ν. The tests are run on simulations of multiple time steps over 91 daily time tests to test the quality of the resulting terminal distribution. 91 days corresponds to a quarterly time interval. The tests are summarized in Table 1. Table 2 contains a second set of tests run for a much longer time horizon, 3,650 days, or 10 years. In both sets of tests 1,000,000 simulation paths are run. The computing times in Table 1 are reported for simulations run separately on a CPU and on a graphical processing unit (GPU) to show the benefits of parallel processing with a GPU. The initial seeds for each simulation path on the GPU are set using the step-ahead method, so that the GPU simulations calculated with double precision match exactly the CPU simulations that are run sequentially. Simulation on a GPU runs much faster if single precision arithmetic is used and the single precision functions in the Cuda C Math Library are used. Table 1 includes computing times for both double precision and single precision GPU calculations. The tests in Table 2 are run on a GPU only because the computing times are much longer on a CPU without parallel processing. The same tests were applied to simulations using the exact methods for simulating non-central chi squared variates described in section 2. The exact simulations, one set using small daily time steps and one using a single time step over the full time period (91 days or 10 years), produce simulations that pass all of the statistical tests in Tables 1 and 2. The results for the exact simulation are not reported in Tables 1 and 2. The computing times for the exact method over daily time steps are roughly 5 to 10 times the computing times for the approximation methods. Simulation of the exact method over a single time step is, however, much faster, but most pricing model applications of the square root process require simulation over the full path of the process.

A variety of standard statistical tests are calculated for the simulated distributions. The mean and variance of the simulated distributions are tested with t statistics, which are distributed approximately as standard normal for the large number of simulations. In addition, the Kolmogorov-Smirnov, Cramer-von Mises, and Anderson-Darling statistics are calcu-

lated for the simulated distributions. These statistics, including their large sample distributions, are covered in the following references: Anderson and Darling (1952, 1954), Csorgo and Faraway (1996), and Simard and L'Ecuyer (2011). These 3 statistics are designed to test whether an empirical distribution has been drawn from a specific distribution. The approximation methods are approximations for the non-central chi squared distribution, so that there is an expectation that these tests are likely to show rejection that the simulated distributions are non-central chi squared. The tests serve as measures for the goodness of fit for the different simulation methods; cases where the test statistics are less than the corresponding critical values indicate that the simulated distribution is close to the non-central chi squared distribution. It should be noted that the Anderson-Darling test uses weights based on the underlying distribution, under the null hypothesis, and places more weight on the tails of the distribution. The Anderson-Darling statistic is particularly sensitive to simulations in the tails, including those close to zero; a simulation of a zero value produces an infinite value for the Anderson-Darling statistic. For this reason, there are cases with the QE method and the Euler method that produce infinite Anderson-Darling statistics.

The t tests for means and variances are two sided tests so that the critical values for 1 million simulations are ±1.96 for the 95% level, ±2.58 for the 99% level, and ±3.29 for the 99.9% level. The t test for the mean is standard. The t test for the variance is calculated by dividing the difference between the simulated variance and the variance for the non-central chi squared by the square root of the variance of the simulated variance. The variance of the simulated variance is calculated by using a sample (simulated) estimate of the 4th moment.

$$\frac{E(x(t) - \text{Mean})^4 - \text{Var}^2}{N},$$

where Mean and Var represent the analytic mean and variance for the non-central chi squared, and N is 1 million for the number of simulations. This variance uses a calculation of the 4th moment around the mean for the simulated values. The critical values for the Kolmogorov-Smirnov statistic have been calculated by running the program provided by Simard & L'Ecuyer (2011). The critical values for this statistic, with a sample size of 1 million, are as follows: 0.001358 for p = 0.95, 0.001627 for p = 0.99, and 0.001949 for p = 0.999. The critical values for the Cramer-von Mises tests are from Table 1, p. 229, in Csorgo & Faraway (1996). For a sample size of 1 million, the critical values of the Cramer-von Mises statistic are as follows: 0.4612 for p = 0.95, 0.7430 for p = 0.99, and 1.1616 for p = 0.999. The distribution of the Anderson-Darling statistic depends on the underlying distribution, under null hypothesis. To establish critical values for the Anderson-Darling statistic, the exact non-central chi squared distribution has been simulated to produce repeated simulations of this test statistic. The critical values for this test statistic do vary with the degrees of freedom parameter. The model simulations indicate that the critical values for the cases in Table 1 are generally 2.3 to 2.5 for the 95th percentile and 3.7 to 4.2 for the 99th percentile.

Finally, a few comments regarding the simulation of the models and the calculation of the non-central chi squared distribution are necessary. The QB2Exp, QBNC1, and QE sim-

ulation methods and the Euler method have been programmed in C++. A separate function for the simulation of the model in (2) with rescaling to match the variance has been programmed for the case when the degrees of freedom, γ, are greater than or equal to 1. This model is labeled FastQ. If $\gamma < 1$, the QB2Exp and QBNC1 methods use Andersen's quadratic method that matches the mean and variance with ψ_c set to 2. To improve computing performance, the functions for the different methods have been programmed with upfront calculations for model parameters, where possible. Additional simulations of the exact non-central chi squared distribution have been programmed, but the results are not reported as the exact methodology produces simulations that pass all statistical tests. The non-central chi squared CDF is calculated using the function available in the Boost library. The model simulations for normal, beta, and exponential distributions are based on the inverse method applied using the pseudo RNG for a uniform distribution, MRG32ka, which is a combined multiple recursive RNG developed by L'Ecuyer (1999). This RNG has a very long period and passes the tests for RNG's (Crush and Big Crush) in L'Ecuyer and Simard (2007). It is used in statistical packages (Python, R, Matlab) as well as the Curand library for parallel processing on Nvidia GPU's. The step ahead method for the MRG32k3a RNG has been developed in the online paper by Bradley, du Toit, Giles, Tong, and Woodhams, and this method is used to set starting seeds for the RNG on each processor when running on a GPU. The step ahead methodology ensures that the seeds and the simulations on the GPU match those that are run sequentially on a CPU. The counter based RMG's developed in Salmon, Moraes, Dror, and Shaw (2011) are also potential candidates for use in parallel processing. These RNG's, also known as Philox, pass the tests for RNG's, have step-ahead methods necessary for parallel processing, and are available in the Curand library and most statistical packages. The model calculations and computing times reported in Tables 1 and 2 have been performed with an AMD Ryzen Threadripper PRO 5955WX CPU, which has a maximum turbo frequency of 4.50 gigahertz, and an Nvidia RTX 4070 GPU. Performance results will vary across different processors.

The statistical tests in Table 1 are organized in panels to show the results across different values for the degrees of freedom, ν, and the initial value $x(0)$. As noted, the degrees of freedom are controlled by adjusting the variance parameter, σ. Panels A to D show results when the ν is either 4.0 or 1.0. The QB2Exp, QBNC1, and QE approximations perform well in these cases, as all statistical tests are passed, except for the Anderson-Darling test using the QE method when $\nu = 1$. In these cases, the QE method simulates terminal values that are exactly zero. It should be noted that if one adjusts the switch threshold in the QE method from $\psi_c = 1.5$ to $\psi_c = 2.0$, the QE method will always use the quadratic function of a normal simulation when $\nu \geq 1$, so that zeros are not simulated. The Euler method simulates terminal values of zero, and it fails the t test for the simulated variance when $\nu = 1$ and $x(0) = 0.01$, which is the smaller value for $x(0)$. It should be noted that the Euler method performs well when $\nu > 2$, and it runs faster than the other 2 methods. In summary, the quadratic approximation methods perform well when $\nu \geq 1$. The results for the statistical tests start to deteriorate as ν is decreased below 1 in panels E to J. When ν is either 0.25 or 0.1111, the Euler method produces significant biases for the mean and variance, and it fails all statistical

tests. For $\nu < 1$, the QB2Exp, QBNC1, and QE methods pass the tests for the mean and variance, but these approximation methods fail on the other statistical tests. The goodness of fit statistics for all three approximation methods become worse as the ν parameter decreases. The goodness of fit is worst for the QE method and the Euler method. The QB2Exp approximation method provides a better fit than the QBNC1 method in most cases. The QB2Exp requires more computing time with double precision on a CPU or a GPU, but less time with single precision on a GPU. All methods have been optimized for faster performance with single precision arithmetic on a GPU by using single precision functions in the Cuda C Math Library. The calculations for simulations of the exponential distributions require double precision. The single precision version of the QB2Exp methodology uses the following functions from the Cuda C Math Library () that are designed to run more efficiently: sqrtf, expf, and powf.

Table 2 contains results for simulations over a much longer time horizon, which is set at 3,650 days or 10 years. Panels A to D in this table contain simulation tests for $\nu = 2.0$ and $\nu = 0.64$, with values of $x(0)$ set to 0.04 or 0.01. The QB2Exp, QBNC1, QE, and Euler methods all perform well in the tests when $\nu = 2.0$, with the exception that the Euler method does produce some zero terminal values which result in an infinite value for the Anderson-Darling test statistics. When $\nu = 0.64$, the test statistics indicate failures similar to those observed in Table 1. Table 2 includes the run times on a GPU for both double and single precision. These tests were not run on a CPU as the computing times would have been much longer. The computing times for single precision in Table 2 are less than 2 seconds for the QB2Exp, QBNC1 and QE methods and show that one can run these simulation methods with daily time steps for large numbers of simulations, 1 million, over a long-time horizon, 10 years, with a relatively quick response time.

The statistical tests for the simulated distributions indicate that the approximation methods do not reproduce the non-central chi squared distribution exactly when $\nu < 1$, but the tests alone do not isolate the approximation errors for the simulated distributions. The approximation methods are generally accurate for most of the distribution for the square root process, but there are significant approximation errors in the lower tail when $\nu < 1$. This observation is shown in the plots of the simulated distributions in Figure 3 for the cases when ν is either 0.64 or 0.25. The graphs include plots of the simulated distributions for $x(t)$ versus the exact non-central chi squared using the simulations of the terminal values for 91 days, in which t approximately equal to 0.25. The statistical tests for the goodness of fit compare the simulated distributions for $x(t)$ with the non-central chi squared distribution function. Figure 3 includes plots of the simulated distribution with the non-central chi squared distribution function, which are labeled "Simulated Distributions vs. Non-Central Chi Squared." One can also test the empirical distributions by calculating the simulated values of the distribution function, $F\big(x(t)\big)$, using the non-central chi squared distribution evaluated with the simulations of $x(t)$ for the different approximation methods. The simulated distribution functions should plot along a 45-degree line when sorted. Figure 3 includes these graphs labeled as "Simulated Cumulative Distribution Functions." Additional

graphs are included to show the behavior in the lower tails. The approximation models generally provide good fits across most of the distribution; in most cases, the simulated distributions are indistinguishable from the non-central chi squared distribution when the entire range of the distribution is plotted. The approximation methods perform well across most of the distribution, but there are significant approximation errors in the lower tail. In the graphs of the lower tail, one can see the simulation of zero values with the QE method. There are approximation errors with the QB2Exp method in the lower tail, but it generally provides a much better fit than the QE method.

Table 1. Tests of the Simulation Methods with parameters: $\kappa = 0.25, \theta = 0.04, \lambda = -.125$ using 1,000,000 simulation paths over 91 days with 1 time step per day

Panel A. $x(0) = 0.04, \sigma = 0.1, \nu = 4.0$

	FastQ	QE	Euler	Analytic
Mean	0.041236	0.041236	0.041236	0.041227
t test (Mean)	0.09	0.09	0.09	
Std. Deviation	0.0099068	0.0099068	0.0099068	0.0099090
t test (Variance)	-0.29	-0.29	-0.16	
Kolmogorov-Smirnov	0.0009275	0.0009275	0.0010556	
Cramer-von Mises	0.1158	0.1158	0.1880	
Anderson-Darling	0.8297	0.8297	1.1171	
CPU Run Time (sec.)	2.864	3.327	2.776	
GPU Run Time (sec.)	0.051 / 0.005	0.060 / 0.012	0.040 / 0.003	

Panel B. $x(0) = 0.01, \sigma = 0.1, \nu = 4.0$

	FastQ	QE	Euler	Analytic
Mean	0.012152	0.012152	0.012152	0.012148
t test (Mean)	0.06	0.06	0.06	
Std. Deviation	0.0051767	0.0051767	0.0051749	0.0051775
t test (Variance)	-0.19	-0.19	-0.61	
Kolmogorov-Smirnov	0.0008953	0.0008956	0.0011123	
Cramer-von Mises	0.1107	0.1108	0.2874	
Anderson-Darling	0.7762	0.7762	1.6528	
CPU Run Time (sec.)	2.856	3.324	2.776	
GPU Run Time (sec.)	0.051 / 0.005	0.062 / 0.011	0.043 / 0.003	

Panel C. $x(0) = 0.04, \sigma = 0.2, \nu = 1.0$

	QB2Exp	QE	Euler	Analytic
Mean	0.041244	0.041244	0.041244	0.041227
t test (Mean)	0.12	0.12	0.12	
Std. Deviation	0.019814	0.019814	0.019816	0.019818
t test (Variance)	-0.22	-0.22	-0.10	
Kolmogorov-Smirnov	0.0009495	0.0009495	0.0012406	
Cramer-von Mises	0.1170	0.1170	0.3251	
Anderson-Darling	0.8436	∞	∞	
CPU Run Time (sec.)	2.858	3.326	2.782	
GPU Run Time (sec.)	0.051 / 0.005	0.062 / 0.012	0.043 / 0.003	

Panel D. $x(0) = 0.01, \sigma = 0.2, \nu = 1.0$

	FastQ	QE	Euler	Analytic
Mean	0.012155	0.012155	0.012162	0.012148
t test (Mean)	0.07	0.07	0.14	
Std. Deviation	0.010353	0.010353	0.010343	0.010355
t test (Variance)	-0.15	-0.15	-1.08	
Kolmogorov-Smirnov	0.0009491	0.003415***	0.005012***	
Cramer-von Mises	0.1150	0.1400	0.8718**	
Anderson-Darling	0.8409	∞	∞	
CPU Run Time (sec.)	2.853	3.323	2.776	
GPU Run Time (sec.)	0.051 / 0.005	0.065 / 0.014	0.040 / 0.003	

Panel E. $x(0) = 0.04, \sigma = 0.25, \nu = 0.64$

	QB2Exp	QBNC1	QE	Euler	Analytic
Mean	0.041247	0.041247	0.041247	0.041248	0.041227
t test (Mean)	0.13	0.13	0.13	0.13	
Std. Deviation	0.024768	0.024768	0.024768	0.024770	0.024772
t test (Variance)	-0.21	-0.21	-0.21	-0.13	
Kolmogorov-Smirnov	0.0009531	0.0009531	0.0009531	0.0012770	
Cramer-von Mises	0.1185	0.1185	0.1186	0.4402	
Anderson-Darling	0.8704	0.8666	∞	∞	
CPU Run Time (sec.)	3.512	3.375	3.338	2.828	
GPU Run Time (sec.)	0.068 / 0.005	0.064 / 0.009	0.061 / 0.012	0.039 / 0.004	

Panel F. $x(0) = 0.01, \sigma = 0.25, \nu = 0.64$

	QB2Exp	QBNC1	QE	Euler	Analytic
Mean	0.012156	0.012156	0.012156	0.012201	0.012148
t test (Mean)	0.07	0.07	0.07	0.46	
Std. Deviation	0.012941	0.012940	0.012941	0.012904	0.012944
t test (Variance)	-0.18	-0.17	-0.19	-2.51**	
Kolmogorov-Smirnov	0.001945**	0.002366***	0.03070***	0.02909***	
Cramer-von Mises	0.2357	0.2663	11.063***	40.486***	
Anderson-Darling	6.818**	12.109**	∞	∞	
CPU Run Time (sec.)	4.024	3.444	3.327	2.825	
GPU Run Time (sec.)	0.314 / 0.013	0.091 / 0.023	0.067 / 0.022	0.039 / 0.004	

Panel G. $x(0) = 0.04, \sigma = 0.4, \nu = 0.25$

	QB2Exp	QBNC1	QE	Euler	Analytic
Mean	0.041255	0.041255	0.041255	0.041336	0.041227
t test (Mean)	0.14	0.14	0.14	0.55	
Std. Deviation	0.039628	0.039628	0.039629	0.039559	0.039636
t test (Variance)	-0.18	-0.18	-0.17	-1.76*	
Kolmogorov-Smirnov	0.002295***	0.009704***	0.05942***	0.04254***	
Cramer-von Mises	0.3224	2.3451***	70.656***	53.361***	
Anderson-Darling	6.785**	258.14**	∞	∞	
CPU Run Time (sec.)	3.798	3.433	3.301	2.826	
GPU Run Time (sec.)	0.218 / 0.010	0.093 / 0.019	0.063 / 0.019	0.040 / 0.003	

Panel H. $x(0) = 0.01, \sigma = 0.4, \nu = 0.25$

	QB2Exp	QBNC1	QE	Euler	Analytic
Mean	0.012162	0.012154	0.012162	0.012732	0.012148
t test (Mean)	0.10	0.04	0.10	4.06***	
Std. Deviation	0.020695	0.020696	0.020695	0.020572	0.020710
t test (Variance)	-0.41	-0.40	-0.41	-3.83***	
Kolmogorov-Smirnov	0.007771***	0.04484***	0.2639***	0.1985***	
Cramer-von Mises	9.525***	208.31***	6174.3***	5887.0***	
Anderson-Darling	73.15*	5659.9**	∞	∞	
CPU Run Time (sec.)	6.020	3.889	2.979	2.827	
GPU Run Time (sec.)	0.698 / 0.027	0.139 / 0.040	0.077 / 0.034	0.040 / 0.004	

Panel I. $x(0) = 0.04, \sigma = 0.6, \nu = 0.1111$

	QB2Exp	QBNC1	QE	Euler	Analytic
Mean	0.041265	0.041262	0.041264	0.042089	0.041227
t test (Mean)	0.15	0.14	0.15	3.53***	
Std. Deviation	0.059429	0.059427	0.059429	0.059058	0.059454
t test (Variance)	-0.29	-0.31	-0.29	-4.58***	
Kolmogorov-Smirnov	0.004956***	0.06094***	0.2862***	0.2054***	
Cramer-von Mises	3.560***	488.2***	7818.3***	4461.5***	
Anderson-Darling	25.78**	10428.0**	∞	∞	
CPU Run Time (sec.)	4.543	3.687	3.011	2.827	
GPU Run Time (sec.)	0.438 / 0.018	0.136 / 0.030	0.071 / 0.028	0.040 / 0.003	

Panel J. $x(0) = 0.01, \sigma = 0.6, \nu = 0.1111$

	QB2Exp	QBNC1	QE	Euler	Analytic
Mean	0.012156	0.012156	0.012156	0.014469	0.012148
t test (Mean)	0.05	0.05	0.05	13.17***	
Std. Deviation	0.031006	0.031006	0.031005	0.031566	0.031065
t test (Variance)	-0.77	-0.77	-0.78	-6.53***	
Kolmogorov-Smirnov	0.008191***	0.1192***	0.5537***	0.4076***	
Cramer-von Mises	17.79**	3625.1***	56628.9***	35017.0***	
Anderson-Darling	72.65**	42261.9**	∞	∞	
CPU Run Time (sec.)	6.668	4.338	2.287	2.825	
GPU Run Time (sec.)	0.767 / 0.030	0.147 / 0.043	0.078 / 0.037	0.040 / 0.004	

Notes: significance levels are indicated with asterisks: * for the 95% level, ** for the 99% level, and *** for the 99.9% level.

Table 2. Tests of the Simulation Methods with parameters: $\kappa = 0.25, \theta = 0.04, \lambda = -.125$ using 1,000,000 simulation paths over 3,650 days (10 years) with 1 time step per day

Panel A. $x(0) = 0.04, \sigma = 0.14142136, \nu = 2.0$

	FastQ	QE	Euler	Analytic
Mean	0.068583	0.068583	0.068591	0.068540
t test (Mean)	0.17	0.17	0.20	
Std. Deviation	0.067660	0.067660	0.067668	0.067575
t test (Variance)	0.92	0.92	1.00	
Kolmogorov-Smirnov	0.0007577	0.0007598	0.0007849	
Cramer-von Mises	0.0753	0.0754	0.0877	
Anderson-Darling	0.4989	0.4983	∞	
GPU Run Time (sec.)	1.755 / 0.115	2.197 / 0.425	1.513 / 0.114	

Panel B. $x(0) = 0.01, \sigma = 0.14142136, \nu = 2.0$

	FastQ	QE	Euler	Analytic
Mean	0.059978	0.059978	0.059991	0.059945
t test (Mean)	0.14	0.14	0.19	
Std. Deviation	0.059955	0.059955	0.059963	0.059876
t test (Variance)	0.93	0.93	1.02	
Kolmogorov-Smirnov	0.0006372	0.0006383	0.0008228	
Cramer-von Mises	0.0589	0.0589	0.0851	
Anderson-Darling	0.4326	0.4322	∞	
GPU Run Time (sec.)	1.750 / 0.114	2.191 / 0.425	1.508 / 0.115	

Panel C. $x(0) = 0.04, \sigma = 0.25, \nu = 0.64$

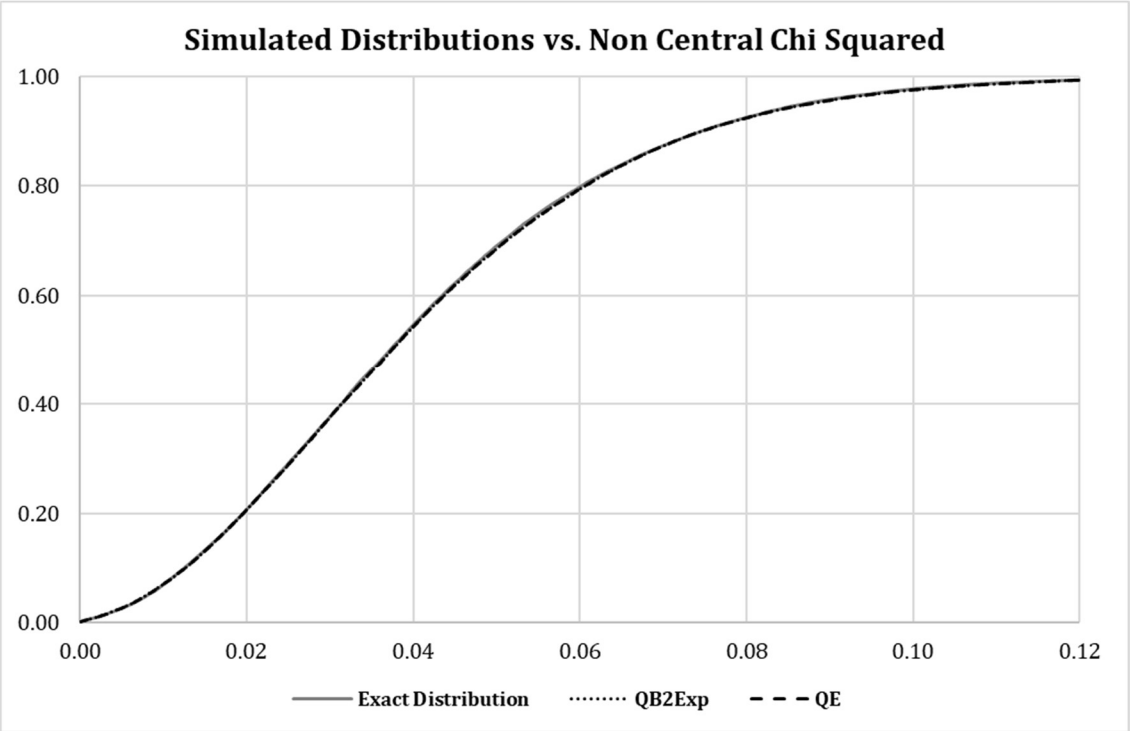
	QB2Exp	QBNC1	QE	Euler	Analytic
Mean	0.068593	0.068596	0.068594	0.070743	0.068540
t test (Mean)	0.15	0.16	0.16	6.37^{***}	
Std. Deviation	0.119757	0.119771	0.119760	0.120551	0.119457
t test (Variance)	1.14	1.19	1.15	4.16^{***}	
Kolmogorov-Smirnov	0.002230^{***}	0.002981^{***}	0.03740^{***}	0.03949^{***}	
Cramer-von Mises	0.2762	0.3417	19.800^{***}	389.07^{***}	
Anderson-Darling	8.689^{**}	18.158^{**}	∞	∞	
GPU Run Time (sec.)	25.678 / 0.889	5.048 / 1.530	3.021 / 1.309	1.530 / 0.114	

Panel D. $x(0) = 0.01, \sigma = 0.25, \nu = 0.64$

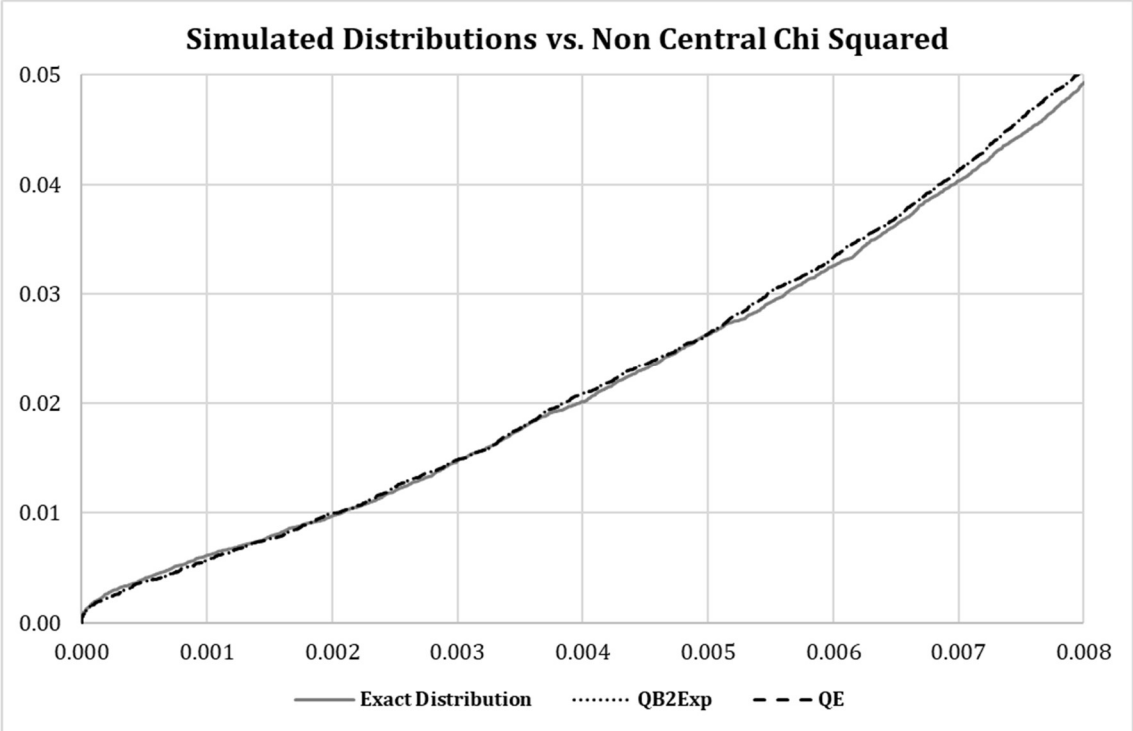
	QB2Exp	QBNC1	QE	Euler	Analytic
Mean	0.059991	0.059991	0.059992	0.062574	0.059945
t test (Mean)	0.14	0.14	0.14	8.08^{***}	
Std. Deviation	0.106126	0.106119	0.106132	0.107660	0.105847
t test (Variance)	1.15	1.12	1.18	7.55^{***}	
Kolmogorov-Smirnov	0.002338^{***}	0.003030^{***}	0.03920^{***}	0.04147	
Cramer-von Mises	0.3292	0.3772	22.7842^{***}	474.59^{***}	
Anderson-Darling	9.800^{**}	19.622^{**}	∞	∞	
GPU Run Time (sec.)	29.322 / 1.032	5.398 / 1.678	3.115 / 1.419	1.529 / 0.115	

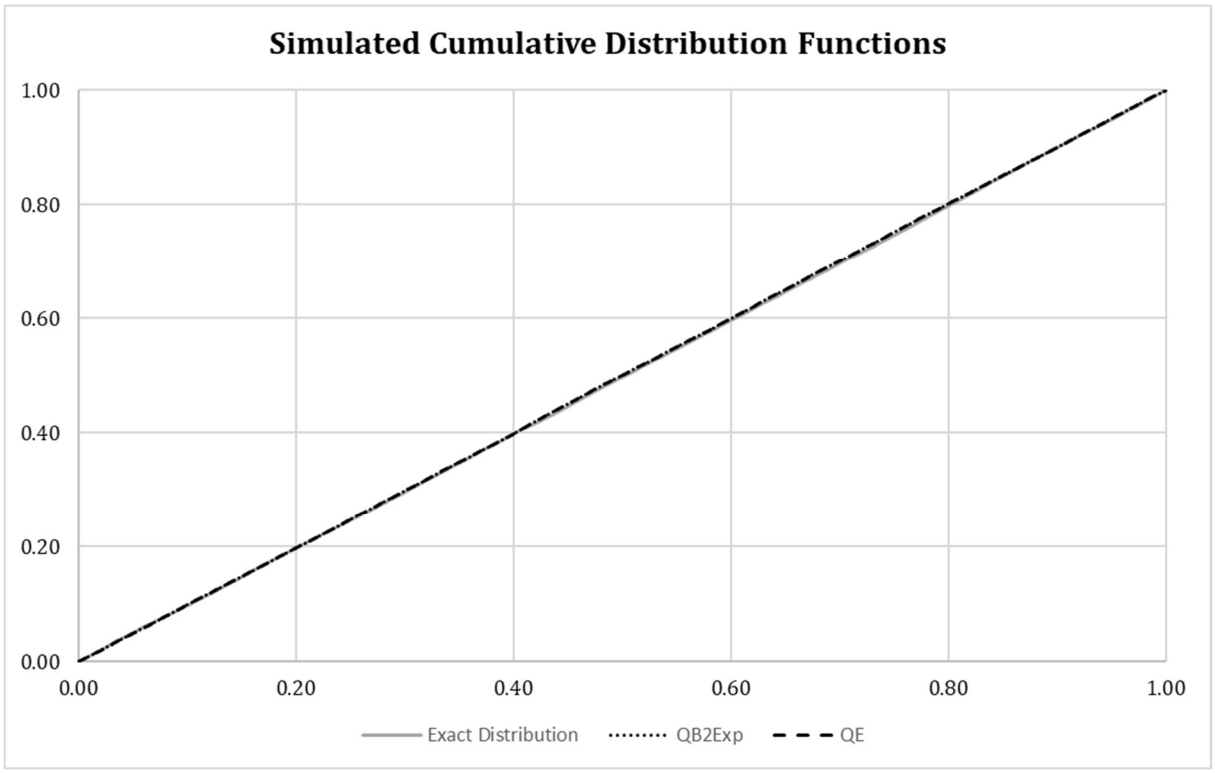
Figure 3. Simulated Distributions

$x(0) = 0.04, \sigma = 0.25, \nu = 0.64$

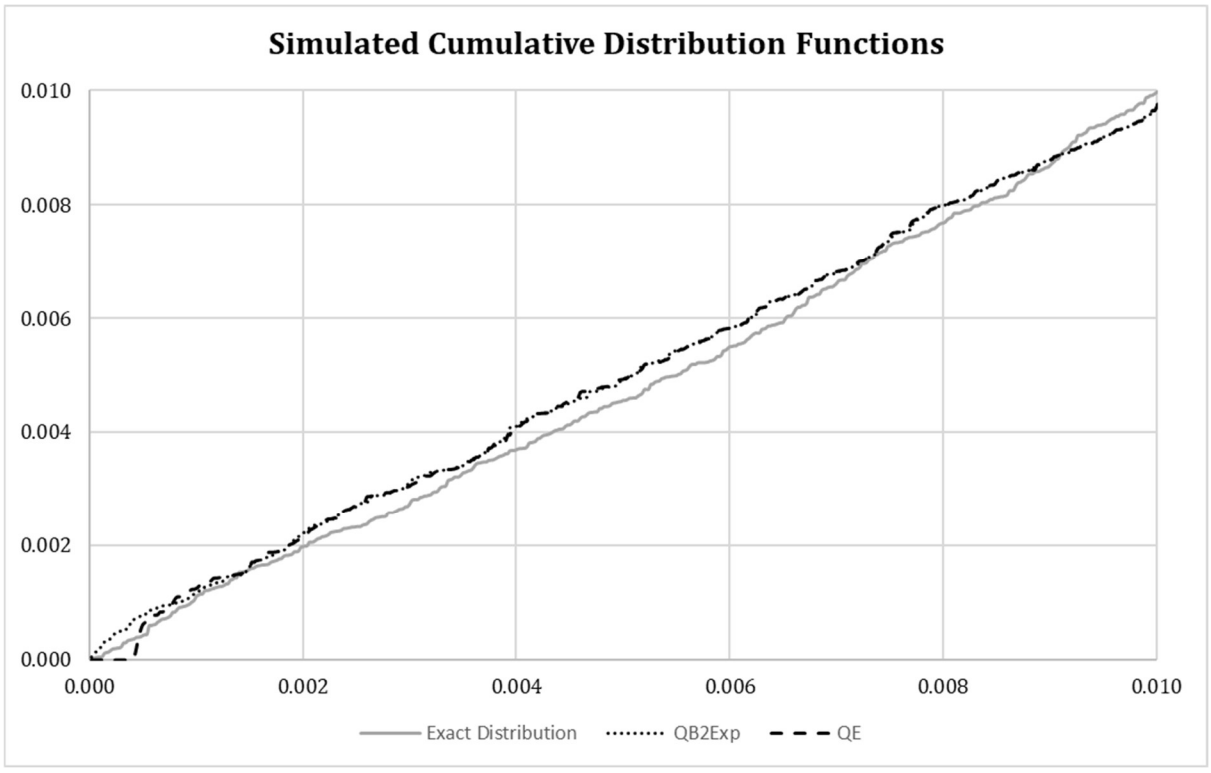


Lower Tail

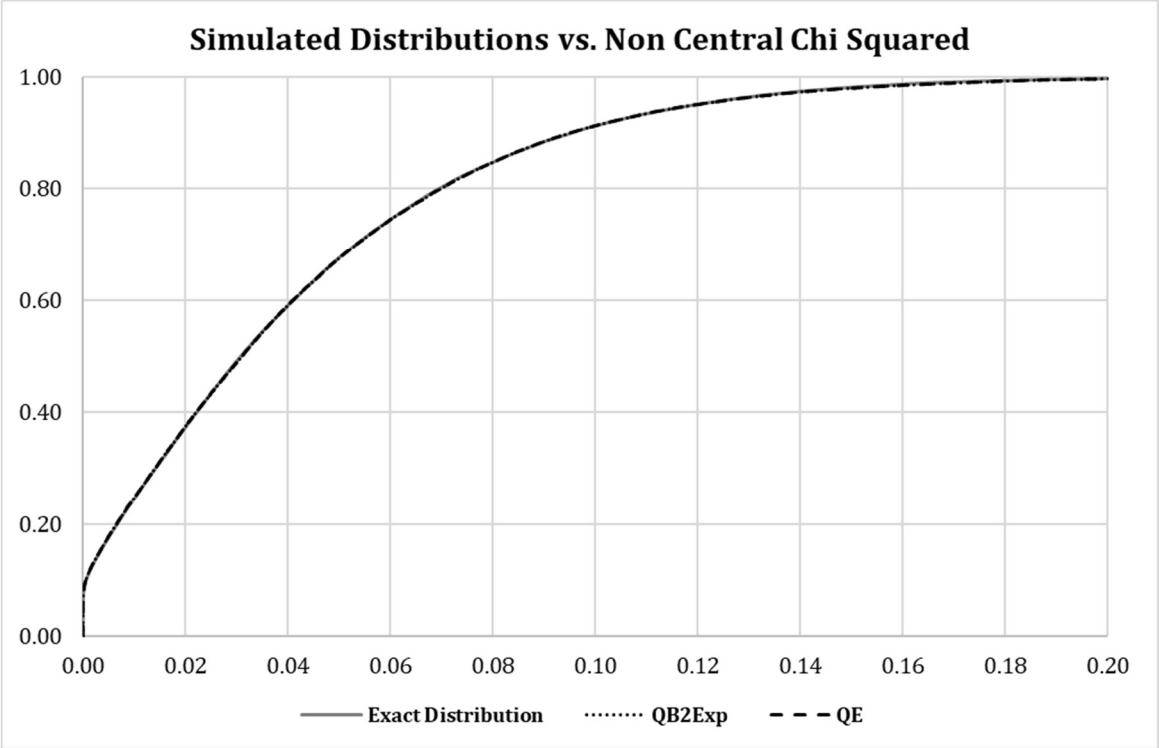



Simulated Cumulative Distribution Functions
1.00
0.80
0.60
0.40
0.20
0.00
0.00
0.20
0.40
0.60
0.80
1.00
Exact Distribution
QB2Exp
QE

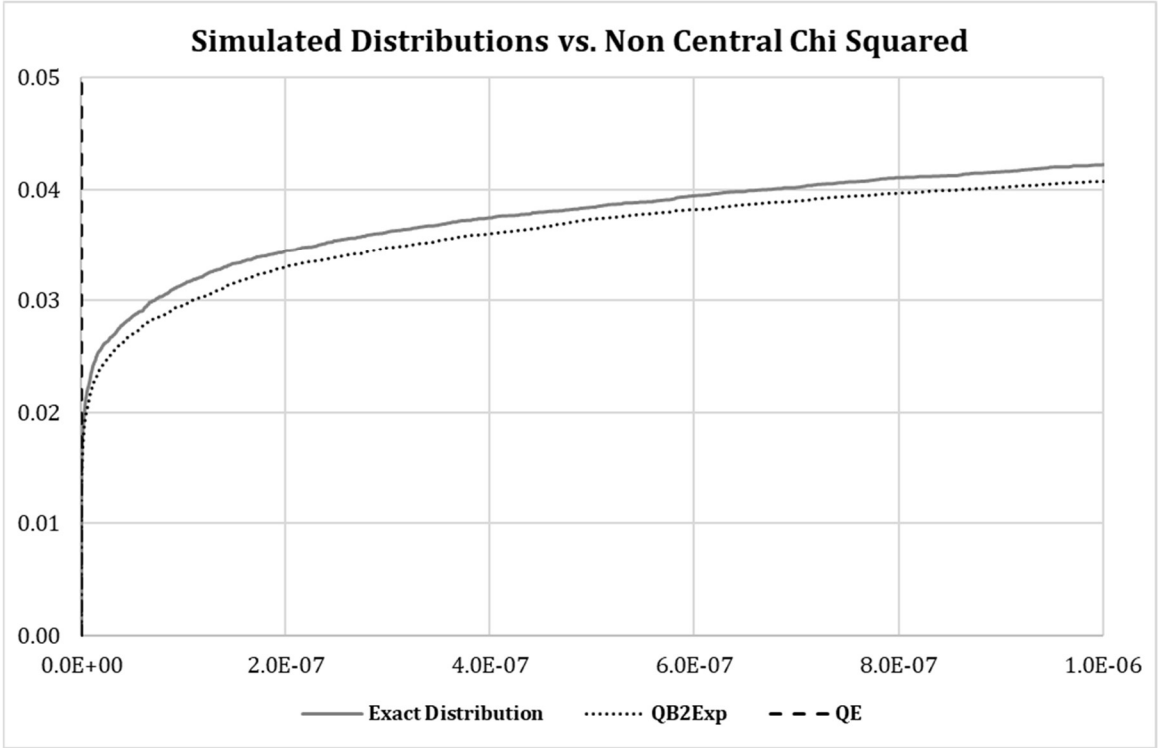
Lower Tail

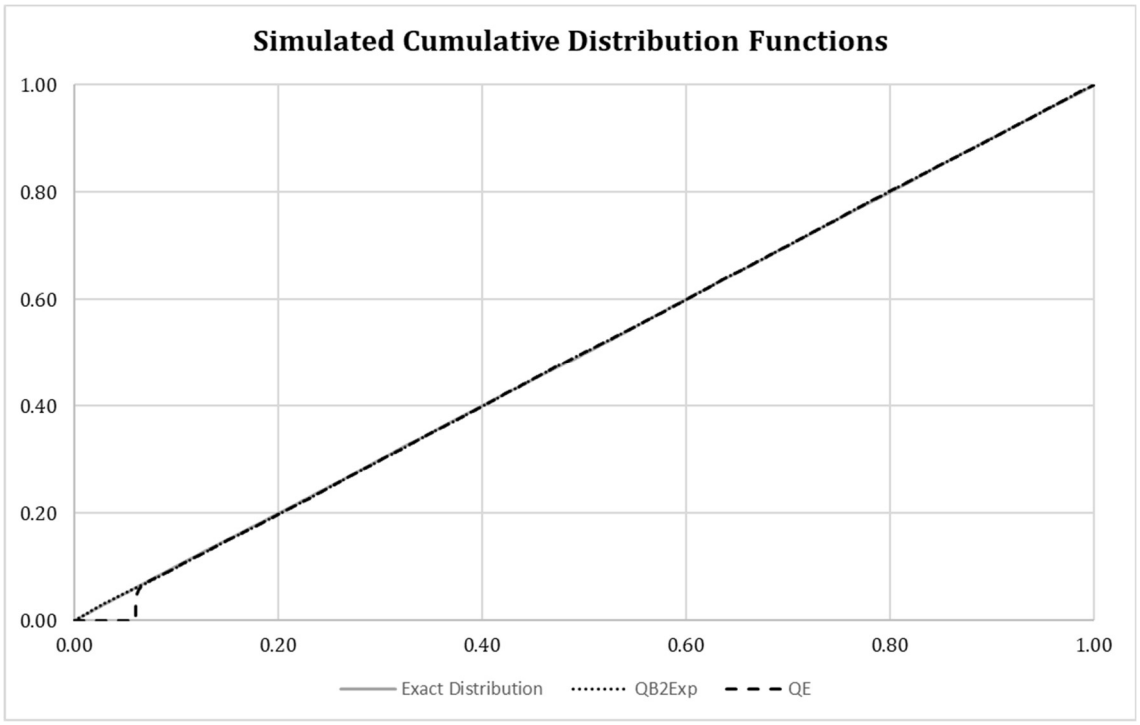

Simulated Cumulative Distribution Functions
0.010
0.008
0.006
0.004
0.002
0.000
0.000
0.002
0.004
0.006
0.008
0.010
Exact Distribution
QB2Exp
QE

$x(0) = 0.04, \sigma = 0.4, \nu = 0.25$



Lower Tail




Simulated Cumulative Distribution Functions
1.00
0.80
0.60
0.40
0.20
0.00
0.00
0.20
0.40
0.60
0.80
1.00
Exact Distribution
QB2Exp
QE

Lower Tail

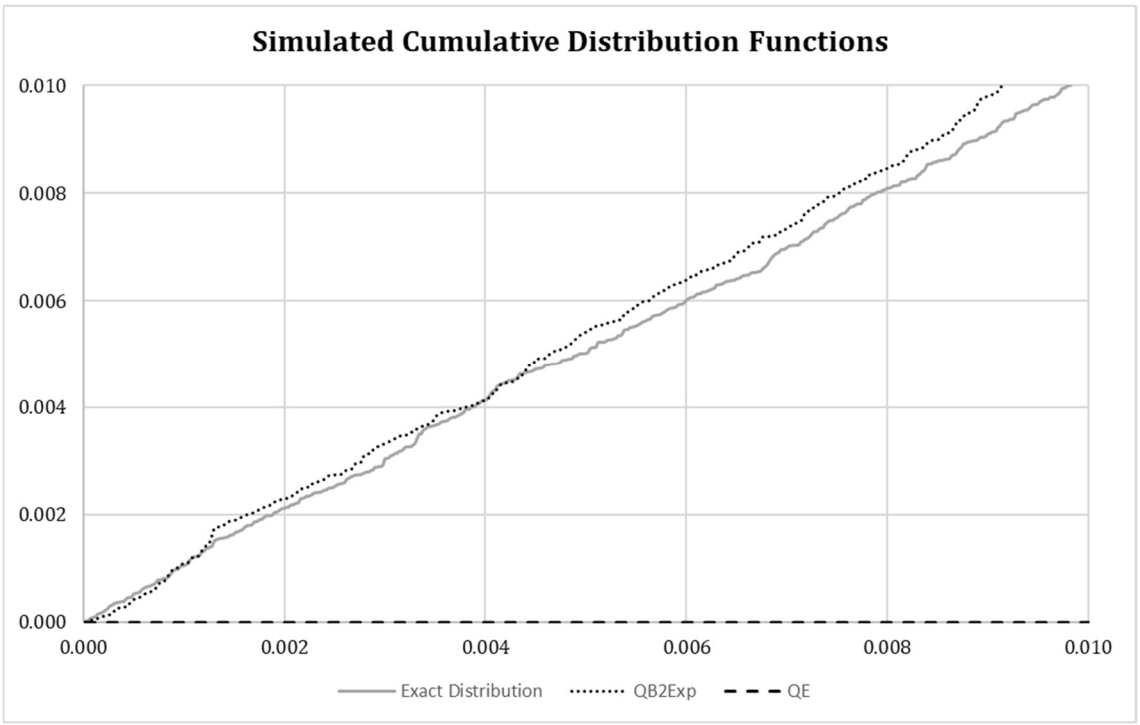

Simulated Cumulative Distribution Functions
0.010
0.008
0.006
0.004
0.002
0.000
0.000
0.002
0.004
0.006
0.008
0.010
Exact Distribution
QB2Exp
QE

Table 3 contains additional tests of the convergence of simulated option prices in the Heston model. The Heston model is a stochastic volatility model with the variance process modeled as a square root diffusion.

$$d\log S = \left(r - \tfrac{1}{2}v\right)dt + \sqrt{v}\,dz_1$$

$$dv = (\kappa\theta - (\kappa+\lambda)v)\,dt + \sigma\sqrt{v}\,dz_2$$

The model includes a correlation between the Brownian motions: $E(dz_1 dz_2) = \rho\,dt$. The model is simulated by simulating a square root diffusion for v and using the following stochastic integral for the stock price to capture the correlation.

$$\log S(t) = \log S(0) + \int_0^t \left(r - \tfrac{1}{2}v(s)\right)ds + \int_0^t \sqrt{v(s)}\left(\sqrt{(1-\rho^2)}\,dz_1(s) + \rho\,dz_2(s)\right)$$

The simulation for $v(t)$ is decomposed as follows.

$$v(t) = e^{-(\kappa+\lambda)t}v(0) + \kappa\theta\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right) + \sigma\int_0^t \sqrt{v(s)}\,dz_2(s)$$

and

$$\int_0^t \sqrt{v(s)}\,dz_2(s) = \frac{\left(v(t) - e^{-(\kappa+\lambda)t}v(0) - \kappa\theta\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)\right)}{\sigma}$$

At each time step, the log of the stock price is simulated as follows.

$$\log S(t) = \log S(0) + \left(r - \tfrac{1}{4}(v(0)+v(t))\right)t + \int_0^t \rho\sqrt{v(s)}\,dz_2(s) + \sqrt{(1-\rho^2)v(0)t}\,\tilde{Z}$$

where $\tilde{Z}$ is a standard normal simulation.

The simulations in Table 3 are simulations of the option prices for strikes from 70 to 125, with the initial stock price at 100. The parameters for the square root process are included in the table. Again, the degrees of freedom are adjusted by resetting the σ parameter. The interest rate has been set at 3.0% per annum and the correlation parameter has been set at -0.90 to capture the negative correlation typically observed between volatility and stock prices. Again, 1 million simulation paths are run for a time horizon of 91 days (3 months), and the time steps per day are set at 2. Initial runs with only 1 time step per day resulted in some biased simulations for all methods, including the exact simulation of the non-central chi squared. The table includes one case with the degrees of freedom, ν, set to 2.0, and all methods, including the Euler method perform well in that the option prices are converging to the analytic solution for the Heston model. The analytic solutions for the option prices in the Heston model are computed separately by using the Fourier inversion method to calculate the relevant probability functions. Table 3 includes 3 additional cases

for the degrees of freedom, ν = 0.8889, 0.5, and 0.08. The tests in the table show that both the QB2Exp and the QE methods perform well and generally converge as the degrees of freedom are decreased to 0.08. In sharp contrast, the Euler method has significant biases and does not converge for many of the option strikes when the degrees of freedom are less than one. Additional tests have been performed by increasing the number of time steps per day and the number of simulation paths to test the convergence of the simulation methods. When the degrees of freedom are less than 1, the QB2Exp and QE methods are converging to the analytic solutions, and the Euler method produces significant biases.

The last set of tests include simulations of an integral of a square root process to approximate the discount function in the Cox-Ingersoll-Ross (CIR) model. The integrals are approximated using daily time steps over 1-year and 5-year time periods. The simulated discount functions should converge to the analytic solution in the CIR model. The results are summarized in Table 4 for two sets of parameter values, one with degrees of freedom at 2.0 and one with degrees of freedom at 0.50. The tests are performed with 1,000,000 simulation paths and the QB2Exp and QE methods show convergence to the corresponding analytic solutions. The Euler method produces simulations that convergence when the degrees of freedom are 2.0, but the simulations with degrees of freedom at 0.50 do not converge.

Table 3. Simulation Tests for Heston Model with $\kappa = 2.0, \theta = 0.04, \lambda = -.50, \rho = -0.90$, r = 0.03, using 1,000,000 simulation paths over 91 days with 2 time steps per day

Panel A. $x(0) = 0.04, \sigma = 0.4, \nu = 2.0$

		Heston	QB2Exp	t test	QE	t test	Euler	t test
Put	70	0.0301	0.0298	-0.43	0.0298	-0.44	0.0297	-0.62
Put	75	0.0828	0.0827	-0.06	0.0827	-0.06	0.0826	-0.18
Put	80	0.2073	0.2076	0.19	0.2076	0.19	0.2077	0.22
Put	85	0.4761	0.4771	0.42	0.4771	0.42	0.4775	0.63
Put	90	1.0090	1.0106	0.48	1.0106	0.48	1.0121	0.94
Put	95	1.9823	1.9857	0.71	1.9857	0.71	1.9887	1.35
Put	100	3.6206	3.6249	0.66	3.6249	0.66	3.6294	1.37
Call	100	4.3658	4.3607	-1.00	4.3607	-1.00	4.3645	-0.25
Call	105	1.9369	1.9334	-1.07	1.9334	-1.07	1.9373	0.14
Call	110	0.5451	0.5440	-0.68	0.5440	-0.68	0.5463	0.79
Call	115	0.0660	0.0667	1.56	0.0667	1.57	0.0671	2.41
Call	120	0.00288	0.00289	0.07	0.00289	0.08	0.00292	0.46
Call	125	0.00007	0.00007	0.10	0.00007	0.10	0.00007	0.18

Panel B. $x(0) = 0.04, \sigma = 0.6, \nu = 0.8889$

		Heston	QB2Exp	t test	QE	t test	Euler	t test
Put	70	0.0639	0.0637	-0.31	0.0637	-0.31	0.0635	-0.56
Put	75	0.1429	0.1432	0.22	0.1432	0.22	0.1430	0.11
Put	80	0.2998	0.3003	0.27	0.3003	0.27	0.3005	0.38
Put	85	0.5945	0.5960	0.52	0.5960	0.52	0.5969	0.84
Put	90	1.1206	1.1227	0.54	1.1227	0.54	1.1253	1.21
Put	95	2.0170	2.0210	0.78	2.0210	0.77	2.0261	1.74
Put	100	3.4827	3.4873	0.68	3.4873	0.68	3.4964	2.02
Call	100	4.2278	4.2232	-1.08	4.2232	-1.08	4.1917	-8.39
Call	105	1.5922	1.5897	-0.99	1.5897	-0.99	1.5672	-10.11
Call	110	0.2490	0.2491	0.10	0.2491	0.11	0.2445	-5.04
Call	115	0.0138	0.0140	1.10	0.0140	1.09	0.0140	0.79
Call	120	0.00057	0.00053	-1.12	0.00053	-1.12	0.00053	-0.91
Call	125	0.00002	0.00002	-0.15	0.00002	-0.15	0.00002	0.09

Panel C. $x(0) = 0.04, \sigma = 0.8, \nu = 0.50$

		Heston	QB2Exp	t test	QE	t test	Euler	t test
Put	70	0.1046	0.1046	0.04	0.1046	0.04	0.1045	-0.10
Put	75	0.2049	0.2054	0.33	0.2054	0.33	0.2057	0.45
Put	80	0.3830	0.3840	0.42	0.3840	0.42	0.3851	0.91
Put	85	0.6871	0.6891	0.62	0.6891	0.63	0.6923	1.62
Put	90	1.1895	1.1923	0.64	1.1923	0.65	1.2002	2.48
Put	95	1.9983	2.0027	0.77	2.0027	0.77	2.0207	3.97
Put	100	3.2883	3.2934	0.72	3.2936	0.74	3.3369	6.80
Call	100	4.0335	4.0294	-1.10	4.0292	-1.13	3.8965	-37.98
Call	105	1.2265	1.2259	-0.30	1.2258	-0.35	1.1616	-33.59
Call	110	0.1130	0.1136	0.96	0.1135	0.88	0.1103	-4.38
Call	115	0.0069	0.0072	1.55	0.0072	1.56	0.0071	0.99
Call	120	0.00046	0.00050	0.94	0.00050	0.93	0.00051	1.04
Call	125	0.00003	0.00005	1.03	0.00005	1.03	0.00005	1.10

Panel D. $x(0) = 0.04, \sigma = 2.0, \nu = 0.08$

		Heston	QB2Exp	t test	QE	t test	Euler	t test
Put	70	0.2907	0.2909	0.10	0.2911	0.16	0.3462	20.14
Put	75	0.4139	0.4144	0.16	0.4145	0.20	0.5041	26.39
Put	80	0.5784	0.5791	0.19	0.5790	0.16	0.7222	34.34
Put	85	0.7966	0.7976	0.22	0.7973	0.16	1.0244	44.82
Put	90	1.0890	1.0905	0.27	1.0900	0.19	1.4519	59.31
Put	95	1.4989	1.5013	0.36	1.5006	0.26	2.0899	80.62
Put	100	2.1743	2.1778	0.47	2.1772	0.38	3.2082	117.91
Call	100	2.9194	2.9183	-0.47	2.9186	-0.34	2.4764	-202.14
Call	105	0.2656	0.2677	1.84	0.2678	1.98	0.3334	53.34
Call	110	0.0386	0.0388	0.36	0.0387	0.26	0.0470	15.29
Call	115	0.0087	0.0089	0.76	0.0088	0.56	0.0099	4.51
Call	120	0.00225	0.00231	0.44	0.00229	0.26	0.00250	1.70
Call	125	0.00063	0.00068	0.62	0.00067	0.48	0.00076	1.49
Call	130	0.00019	0.00054	0.65	0.00053	0.54	0.00061	1.52

Table 4. Simulation Tests for the Discount in the CIR Model

Parameters: $\kappa = 0.5, \theta = 0.04, \lambda = -.25, x(0) = 0.04$
1,000,000 simulation paths with 1 time step per day

Panel A. $\sigma = 0.20, \nu = 2.0$

Years	Analytic	Fast Quadratic	t test	QE	t test	Euler	t test
1	0.95659608	0.95660239	0.30	0.95660239	0.30	0.95660233	0.30
5	0.76412413	0.76427294	1.15	0.76427299	1.15	0.76424651	0.95

Panel B. $\sigma = 0.40, \nu = 0.50$

Years	Analytic	QB2Exp	t test	QBNC1	t test	QE	t test
1	0.95724993	0.95725349	0.06	0.95725523	0.13	0.95725294	0.07
5	0.79221829	0.79232113	0.52	0.79239516	0.89	0.79233675	0.60

Years	Analytic	Euler	t test
1	0.95724993	0.95688211	-12.44
5	0.79221829	0.78150273	-72.61

4. Summary

There are numerous alternative methods for simulating square root processes. One can always simulate a square root process using the exact simulation method for a non-central chi squared variate, but this method is not well suited for applications in dynamic financial models, such as hedging, parameter calibration, or parallel processing, and it is much slower than alternative methods. The exact simulation of the non-central chi squared requires acceptance-rejection methods in most cases so that the number of simulations over each time step vary and it is difficult to control the simulation with typical RNG's across parallel processing. Several alternative methods, namely the QB2Exp, QBNC1, and QE approximation methods, produce good approximations for the simulation over small time steps and these methods can be easily applied for hedging and parameter calibration, with parallel processing. The various tests in section 3 reveal significant biases if the Euler method is used when the degrees of freedom parameter is less than one. If the degrees of freedom parameter is greater than or equal to one, one should use the quadratic method, which is embedded in the QE, QBNC1, QB2Exp approximations, or the implicit Milstein methods. In these cases, the quadratic method is almost as fast as the Euler method, and it meets the requirement that it never produces negative values. When the degrees of freedom parameter is less than one, the QB2Exp, QBNC1, and QE methods produce simulations that approximate the square root process over small time intervals and converge for pricing applications. It has been noted that the QE method does produce simulations with zero values, whereas the QB2Exp and the QBNC1 methods and the exact simulation method do not. The generation of zero values leads to larger approximations errors for the QE method in the lower tail of the distribution, but otherwise the QE method performs well for pricing applications examined in Tables 3 and 4. In the cases where the QE method produces simulations at zero, the QB2Exp, QBNC1, and exact methods generate simulations close to, but above, zero. The ultimate choice for a simulation method should be based on both computing time and the quality of the approximation. When the degrees of freedom, γ, are greater than or equal to 1, the quadratic methods and the Euler method produce good approximations for the distribution and are quite fast for computing. The Euler method is faster than the quadratic methods. The quadratic method in (2) with rescaling to match the mean and variance is marginally faster than Andersen's quadratic method. When the degrees of freedom, γ, are less than 1, there are several options and the quality of the approximations vary significantly. Relative to the other approximation methods, the QB2Exp method produces distributions that are closer to the exact distribution for square root processes, with relatively fast computing times.

References

Andersen, L.B.G. (2008), "Simple and efficient simulation of the Heston stochastic volatility model," *The Journal of Computational Finance* 11 (3): 1–42

Andersen, L.B.G., and Brotherton-Ratcliffe, R. (2005), "Extended Libor Market Models with Stochastic Volatility," *The Journal of Computational Finance* 9 (1): 1-40.

Andersen, L., P. Jäckel, C. Kahl, "Simulation of Square-Root Processes," chapter in *Encyclopedia of Quantitative Finance*, May 2010, available online at https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470061602.eqf13009

Anderson, T.W., and D. A. Darling (1952), "Asymptotic Theory of Certain "Goodness of Fit" Criteria Based on Stochastic processes," *Annals of Mathematical Statistics* 23: 193–212.

Anderson, T.W., and D. A. Darling (1954), "A Test of Goodness-of-Fit," *Journal of the American Statistical Association* 49: 765–769.

Boost C++ Library: https://www.boost.org/

Bradley, T., J. du Toit, M. Giles, R. Tong, and P. Woodhams, "Parallelisation Techniques for Random Number Generators," available online at https://www.nag.com/IndustryArticles/gpu_gems_article.pdf

Cheng, R.C.H., and G. M. Feast (1979), "Some Simple Gamma Variate Generators," *Journal of the Royal Statistical Society*, Series C (*Applied Statistics*), Vol. 28, No. 3 (1979): 290-295.

Cox, J., J. Ingersoll and S.A. Ross (1985), "A theory of the term structure of interest rates," *Econometrica* 53 (2): 385-407

Cramer, von Mises

Csorgo, S. and J.J. Faraway (1996), "The Exact and Asymptotic Distributions of Cramer-von Mises Statistics," *Journal of the Royal Statistical Society B* 58 (1); 221-234.

Duffie, D., J. Pan and K. Singleton (2000), "Transform analysis and asset pricing for affine jump diffusions," *Econometrica* 68: 1343-1376.

Dyer, J.S., and S. A. Dyer (2008), "Approximations to Inverse Error Functions," *IEEE Instrumentation & Measurement Magazine*, October 2008: 32-36.

1094-6969/08/$25.00©2008IEEE

Feller, W. (1950), "Two Singular Diffusion Problems*," *Annals of Mathematics* 54 (1): 173-182.

Glasserman, P. (2003), *Monte Carlo Methods in Financial Engineering*, Springer Verlag, New York.

Heston, S.L. (1993), "A closed-form solution for options with stochastic volatility with applications to bond and currency options," *Review of Financial Studies* 6 (2): 327-343.

Johnson, N., S. Kotz, and N. Balakrishnan (1995), *Continuous Univariate Distributions*, Vol. 2, 2nd Edition, Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics. John Wiley & Sons, Inc., New York.

Kahl, C. and P. Jäckel (2006), "Fast strong approximation Monte-Carlo schemes for stochastic volatility models", *Journal of Quantitative Finance*, 6 (6): 513-536

Kiesel, R., and M. Lutz (2011), "Efficient pricing of constant maturity swap spread options in a stochastic volatility LIBOR market model," *The Journal of Computational Finance* 14 (4): 37-0.

L'Ecuyer, P. (1999), "Good Parameters and Implementations for Combined Multiple Recursive Random Number Generators," *Operations Research* 47(1):159-164.

L'Ecuyer, P., and R. Simard, "TestU01: A C library for empirical testing of random number generators," *ACM Trans. Math. Software* 33, August 2007.

Salmon, J. K., M. A. Moraes, R. O. Dror, and D. E. Shaw, "Parallel Random Numbers: As Easy as 1, 2, 3," Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC11), New York, NY: ACM, 2011, available online at http://www.thesalmons.org/john/random123/papers/random123sc11.pdf.

Simard, R. and P. L'Ecuyer (2011), "Computing the Two-Sided Kolmogorov-Smirnov Distribution," *Journal of Statistical Software* 39 (11): 1-18.

Wallace, N.D. (1974), "Computer Generation of Gamma Random Variates with Non-integral Shape Parameters," *Communications of the Association for Computing Machinery* 17 (12), December 1974

Wilson, E. B., and M. M. Hilferty (1931), "The Distribution of Chi-Square." *Proceedings of the National Academy of Sciences* 17 (12): 684–688.

Wu, L., and F. Zhang (2006), "LIBOR Market Model with Stochastic Volatility," *Journal of Industrial and Management Optimization* 2 (2): 199-227.

Web Link for C++ code:

Appendix

A. Beta – Non-Central Chi Squared Approximation

This method uses the cumulative distribution function specified in (5), for $x \geq 0$.

$$F(cx) = p\ (\min\,[\mathrm{c}x, 1])^{\nu^*/2} + (1-p)\chi^{2\prime}\left(1, \lambda_{nc}\,, {}^{cx}/_{\beta}\right)$$

First, define M and S^2 to be the mean and variance for cx, and define m_1 and m_2 to be the mean and 2nd moment for the Beta distribution as follows:

$$M = c\left(\left(e^{-(\kappa+\lambda)t} - 1\right)x(0) + \kappa\theta\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)\right)$$

$$S^2 = c^2\,\sigma^2\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)\left(e^{-(\kappa+\lambda)t}x(0) + \tfrac{1}{2}\kappa\theta\left(\frac{1-e^{-(\kappa+\lambda)t}}{\kappa+\lambda}\right)\right)$$

$$m_1 = \frac{\nu^*}{\nu^*+2} \quad \text{and} \quad m_2 = \frac{\nu^*}{\nu^*+4}.$$

The solution to match the mean and variance for the process cx is calculated as follows.

$$\lambda_{nc}^* = \frac{\lambda_{nc}^2 + 6\lambda_{nc} + 3}{(\lambda_{nc}+1)^2}$$

$$a_1 = m_1^2\lambda_{nc}^* - m_2,\ a_2 = S^2 + M^2 + m_2 - 2m_1\lambda_{nc}^* M,\ a_3 = M^2(\lambda_{nc}^* - 1) - S^2$$

$$p = \frac{-a_2 + \sqrt{{a_2}^2 - 4a_1a_3)}}{2\,a_1}$$

And calculate p as follows.

If $4a_1a_3$ is small relative to ${a_2}^2$, then calculate the probability p as follows.

For $x = {}^{-4a_1a_3}/_{a_2^2}$ and $|x| \leq 0.0001$,

$$p = \frac{-a_2 + a_2\left(1 + \frac{1}{2}x - \frac{1}{8}x^2 + \frac{3}{48}x^3 - \frac{5}{128}x^4\ \cdots\right)}{2\,a_1} = \frac{a_2\left(\frac{1}{2}x - \frac{1}{8}x^2 + \frac{3}{48}x^3 - \frac{5}{128}x^4\ \cdots\right)}{2\,a_1}$$

$$\beta = \frac{M - pm_1}{(1-p)(\lambda_{nc}+1)}$$

To simulate the mixture distribution, simulate a uniform number, $U_1 = U(0,1)$. If $U_1 \leq p$, choose the Beta distribution; otherwise choose the non-central chi squared distribution and use the inverse for standard normal distribution to simulate the normal random variate. If

the Beta distribution is selected, create a second uniform random number as follows to simulate the Beta variate:

$$U_2 = \frac{U_1}{p} \quad \text{and} \quad x = \frac{U_2^{2/\nu^*}}{c} \ .$$

If the non-central chi squared is selected, $U_1 > p$, calculate the 2nd uniform random number as follows to simulate the normal variate:

$$U_2 = \frac{1 - U_1}{1 - p} \quad \text{and} \quad x = \frac{\beta\left(\sqrt{\lambda_{nc}} + N(U_2)^{-1}\right)^2}{c} \ .$$

Alternatively, one can use a second independent simulation for U_2.

B. Beta – Double Exponential Approximation

This method uses the following approximation of the CDF for a Non-Central Chi Squared distribution.

For $x \leq x_1$:

$$F^*(x) = \frac{e^{-\frac{1}{2}\lambda_{nc}}}{2^{\frac{\nu}{2}}\,\Gamma\left(\frac{\nu}{2}\right)} \frac{2}{\nu} (cx)^{\frac{\nu}{2}} = \frac{2}{\nu} A^* (cx)^{\frac{\nu}{2}}$$

$$f^*(x) = c\, A^* (cx)^{\frac{\nu}{2}-1}$$

with

$$A^* = \frac{e^{-\frac{1}{2}\lambda_{nc}}}{2^{\frac{\nu}{2}}\,\Gamma\left(\frac{\nu}{2}\right)}$$

For $x_1 < x \leq x_2$

$$F^*(x) = (1 - F^*(x_1)\left(1 - e^{-\beta_1(x - x_1)}\right)$$

$$f^*(x) = (1 - F^*(x_1)\, \beta_1\, e^{-\beta_1(x - x_1)}$$

For $x > x_2$

$$F^*(x) = (1 - F^*(x_2)\left(1 - e^{-\beta_2(x - x_2)}\right)$$

$$f^*(x) = (1 - F^*(x_2)\, \beta_2\, e^{-\beta_2(x - x_2)}$$

The mean and the second moment are evaluated by integrating the PDF's over the different ranges for x. The mean is labeled μ, and the second moment is the sum of μ^2 and the variance of Δx, labeled, Var (Δx).

$$\mu = \int_0^\infty x f^*(x)dx = \int_0^{x_1} xcA^*(cx)^{\frac{\nu}{2}-1}dx + \int_{x_1}^{x_2} x(1-F^*(x_1)\,\beta_1\, e^{-\beta_1(x-x_1)}dx + \int_{x_2}^{\infty} x(1-F^*(x)\,\beta_2\, e^{-\beta_2(x-x_2)}dx$$

$$= \frac{A^*(cx_1)^{\frac{\nu}{2}+1}}{c\left(\frac{\nu}{2}+1\right)} + (1-F^*(x_1))\left[-(x_2-x_1)e^{-\beta_1(x_2-x_1)} + \left(\frac{1}{\beta_1}+x_1\right)\left(1-e^{-\beta_1(x_2-x_1)}\right)\right] + \left(1-F^*(x_2)\right)\left(\frac{1}{\beta_2}+x_2\right)$$

$$\mu^2 + \mathrm{Var}(\Delta x) = \int_0^\infty x^2 f^*(x)dx = \int_0^{x_1} x^2 cA^*(cx)^{\frac{\nu}{2}-1}dx + \int_{x_1}^{x_2} x^2(1-F^*(x_1)\,\beta_1\, e^{-\beta_1(x-x_1)}dx + \int_{x_2}^{\infty} x^2(1-F^*(x)\,\beta_2\, e^{-\beta_2(x-x_2)}dx$$

$$= \frac{A^*(cx_1)^{\frac{\nu}{2}+2}}{c^2\left(\frac{\nu}{2}+2\right)} + (1-F^*(x_1))\left[-(x_2-x_1)^2 e^{-\beta_1(x_2-x_1)} + 2\left(\frac{1}{\beta_1}+x_1\right)\left(-(x_2-x_1)e^{-\beta_1(x_2-x_1)} + \frac{1}{\beta_1}\left(1-e^{-\beta_1(x_2-x_1)}\right)\right) + x_1^2\left(1-e^{-\beta_1(x_2-x_1)}\right)\right] + \left(1-F^*(x_2)\right)\left(\frac{2}{\beta_2}\left(\frac{1}{\beta_2}+x_2\right)+x_2^2\right)$$

Note that $F^*(x_2) = = (1-F^*(x_1)\left(1-e^{-\beta_1(x_2-x_1)}\right)$. For a given value of β_1, set β_2 to match the mean as follows.

Calculate $\hat{\beta}_2$ from

$$\frac{1}{\hat{\beta}_2} + x_2 =$$

$$\frac{\left(\mu - \frac{A^*(cx_1)^{\frac{\gamma}{2}+1}}{c\left(\frac{\gamma}{2}+1\right)} - (1 - F^*(x_1))\left[-(x_2 - x_1)e^{-\beta_1(x_2-x_1)} + \left(\frac{1}{\beta_1} + x_1\right)\left(1 - e^{-\beta_1(x_2-x_1)}\right)\right]\right)}{\left(1 - F^*(x_2)\right)}$$

The solution $\hat{\beta}_2$ is substituted into the equation for the second moment, and an iterative search is performed on β_1 to find the combination β_1, β_2 that match both the mean and second moment, or variance. The search procedure uses the partial derivative of the second moment with respect to β_1 as well as the partial derivatives of $F^*(x_2)$ and $\hat{\beta}_2$ with respect to β_1. The simulation is then based on the inversion of the approximation CDF, $F^*(x)$.

If $U \leq F^*(x_1)$, use

$$x = \frac{1}{c}\left(\frac{U\,\gamma}{2\,A^*}\right)^{\frac{2}{\gamma}}$$

If $F^*(x_1) < U \leq F^*(x_2)$, use

$$x = x_1 - \beta_1 \log\left(\frac{1 - F^*(x_1) - U}{1 - F^*(x_1)}\right)$$

If $U > F^*(x_2)$, use

$$x = x_2 - \beta_2 \log\left(\frac{1 - F^*(x_2) - U}{1 - F^*(x_2)}\right)$$

The iterative search for β_1 and β_2 are required only if the uniform simulation, $U > F^*(x_1)$.