May 2020

Numerical Solution of a Class of Stochastic Functional Differential Equations with Financial Applications

Laszlo Nicolai Fertig
University of Wisconsin-Milwaukee

NUMERICAL SOLUTION OF A CLASS OF STOCHASTIC FUNCTIONAL DIFFERENTIAL EQUATIONS WITH FINANCIAL APPLICATIONS

by

Laszlo Fertig

A Thesis Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Master of Science

in Mathematics

at

The University of Wisconsin-Milwaukee

May 2020

ABSTRACT

NUMERICAL SOLUTION OF A CLASS OF STOCHASTIC FUNCTIONAL DIFFERENTIAL EQUATIONS WITH FINANCIAL APPLICATIONS

by

Laszlo Fertig

The University of Wisconsin-Milwaukee, 2020
Under the Supervision of Professor Chao Zhu

After a brief review of the Euler and Milstein numerical schemes and their convergence results for stochastic differential equations (SDEs) and stochastic functional differential equations (SFDEs), the thesis next proposes two specific SFDEs. The classical Euler and Milstein schemes are developed to find the numerical solutions of these SFDEs, which are then compared with the Ornstein-Uhlenbeck and a modified Ornstein-Uhlenbeck processes. These results are further used to build four different but related stochastic models for stock prices. The fitness of these models is analyzed by comparing real market data. The thesis concludes with a numerical study for option pricing for stock models with path dependent volatilities.

Dedicated to my parents who always support my academic and personal endeavor and choices as well as to Jay, Frank and Lars for the special year we had.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Prof. Chao Zhu for giving me the opportunity to work on this interesting project and for all the helpful feedback.

Secondly, I would like to thank Profs. Stockbridge and Spade for being part of my thesis committee.

Thirdly, I would like to thank Prof. Boyd for the great support in these crazy and difficult times. In situations like this it is important to have a leader like her, who helps everybody to navigate through the difficulties everybody faces now.

Lastly, I would like to thank the math department and especially Prof. Willenbring for making this exchange program possible.

1 Introduction

The field of SDEs (stochastic differential equations) is well-known and there is a lot of research going on about this topic. The famous Black-Scholes pricing formula where the underlying stock price is modeled by the Black-Scholes model got Merton and Scholes the Nobel Prize in 1997 and is a milestone for all stock price models. It is still heavily used in practice, but nowadays we see more and more evidence that the Black-Scholes model also has weaknesses. One of those weaknesses is the general weakness of usual SDEs which is that the future state of a model only depends on the current state of the model and not, as an example, on a mix of past data and current data. To get rid of this problem we extend the concept of SDEs to SFDEs (stochastic functional differential equations). This allows us to give more input in our model than just the current data. In comparison to the Black-Scholes model, a lot of SDEs and especially SFDEs do not have a closed or known distribution. Hence, we are going to need numerical schemes to handle these models. Our second chapter exactly deals with this and builds the theoretical foundation for this thesis. We present the Euler scheme for SDEs and SFDEs and examine under what conditions and in which way the Euler scheme converges.

This leads to chapter 3 where we introduce two particular SFDEs which are both closely related to the Ornstein-Uhlenbeck process. We take a look at their Euler and Milstein schemes, how parameter estimation works in these cases and finally analyze whether the distribution of the SFDEs is stable.

Afterwards, in chapter 4, we use one of the previously introduced SFDEs as well as three other SDEs/SFDEs to model stock prices. As before we first show how parameter estimation works. Moreover, we fit the models to real data and examine which model performs the best and what important and necessary properties of stock market models are.

Finally, in the last chapter we use one of the SFDEs from chapter 3 for a path-dependent volatility model similar to the Heston model, present the corresponding Euler schemes and analyze how option pricing works in this setting.

2 Numerical schemes for SDEs and SFDEs

In this chapter we are going to present the Euler and the Milstein scheme for SDEs, the Euler scheme for SFDEs and observe under what conditions the schemes converge and also in which way they converge. We should note that this chapter is mostly based on the paper of Mao [Mao03].

First, we consider the general SDE

$$dX(t) = a(t, X(t))dt + b(t, X(t))dW(t), \quad t \in [0, T],\ X(0) = X_0 = Z, \tag{1}$$

where W is an $m-$dimensional Wiener process, $T > 0, a(\cdot, \cdot) : [0, T] \times \mathbb{R}^n \to \mathbb{R}^n$ and $b(\cdot, \cdot) : [0, T] \times \mathbb{R}^n \to \mathbb{R}^{n \times m}$. For the components of a and b we write $a = (a^1, ..., a^n)$ and $b = (b^1, ..., b^n)$.

Theorem 2.1

Let $T > 0$ and $a(\cdot, \cdot) : [0, T] \times \mathbb{R}^n \to \mathbb{R}^n, b(\cdot, \cdot) : [0, T] \times \mathbb{R}^n \to \mathbb{R}^{n \times m}$ be measurable functions satisfying

$$|a(t, x)| + |b(t, x)| \leq C(1 + |x|); x \in \mathbb{R}^n, t \in [0, T] \tag{2}$$

for some constant C, (where $|b|^2 = \sum |\sigma_{ij}|^2$) and such that

$$|a(t, x) - a(t, y)| + |b(t, x) - b(t, y)| \leq D|x - y|;\ x, y \in \mathbb{R}^n, t \in [0, T] \tag{3}$$

for some constant D. Let Z be a random variable which is independent of of the $\sigma-$ algebra $\mathcal{F}_\infty^{(m)}$ generated by $W_s, s \geq 0$ and such that

$$\mathbb{E}[|Z|^2] < \infty. \tag{4}$$

Then the SDE from equation (1) has a unique t-continuous solution $X(t)(\omega)$ with the property that $X(t)(\omega)$ is adapted to the filtration $\mathcal{F}_t^Z$ generated by Z and $W_s; s \leq t$ and

$$\mathbb{E}\left[\int_0^T |X(t)|^2 dt\right] < \infty. \tag{5}$$

Proof. See [Øks03] Theorem 5.2.1. □

In applications it is often the case that finding an explicit solution for the SDE from equation (1) is difficult, and sometimes it is even impossible. Hence, there is the need for numerical solutions. Therefore, we are next presenting the easiest and most common method for solving SDEs numerically, the Euler scheme.
First, we define $0 = \tau_0 < \tau_1 < ... < \tau_n < ... < \tau_N = T$ which is our time discretization of $[0, T]$ and we also define the equidistant stepsize

$$\Delta := \tau_{n+1} - \tau_n. \tag{6}$$

Then we define the Euler scheme as

$$Y_{n+1} = Y_n + a(\tau_n, Y_n) \cdot \Delta + b(\tau_n, Y_n) \cdot (W_{\tau_{n+1}} - W_{\tau_n}), \quad Y_0 = X_0. \tag{7}$$

Then it holds $Y_n \approx X(\tau_n)$ and the Euler scheme provides us with a way to simulate possible paths of equation (1). We note that for the Euler scheme for SDEs we need to be able to generate

$$\Delta W := W_{\tau_{n+1}} - W_{\tau_n}. \tag{8}$$

Since W is a Wiener process, it holds that these increments are independent Gaussian with

$$\mathbb{E}(\Delta W) = 0, \quad \mathrm{Var}(\Delta W) = \Delta, \tag{9}$$

i.e. $\Delta W \sim \mathcal{N}(0, \Delta)$.

After we presented the Euler scheme, we are now going to look at conditions when the Euler scheme converges and in which way. We shall note that there are different possible regularity conditions, but we choose them so that they correspond strongly to the regularity conditions in the SFDE case as well as to the conditions for Theorem 2.1.

In addition to the conditions from Theorem 2.1 we need one more condition which is

$$|a(s,x) - a(t,x)| + |b(s,x) - b(t,x)| \leq C(1 + |x|)|s - t|^{\frac{1}{2}} \tag{10}$$

for all $s, t \in [0, T]$ and $x \in \mathbb{R}^n$.

We define $Y(t)$ as the continuous-time process by interpolation of our discrete process from equation (7). Mathematically speaking, we define

$$Y(t) := Y_n + \int_{\tau_n}^{t} a(\tau_n, Y_n)ds + \sum_{j=1}^{m} \int_{\tau_n}^{t} b^j(\tau_n, Y_n)dW_s^j \tag{11}$$

for $t \in [\tau_n, \tau_{n+1}], n = 0, 1, ...$

Theorem 2.2

Under the conditions from Theorem 2.1 in addition to condition (10) the Euler scheme converges in the following sense:

$$\mathbb{E}\left(\sup_{0 \leq t \leq T} |X(t) - Y(t)|^2\right) \leq C\Delta, \tag{12}$$

where $C > 0$ is some constant independent of Δ, but may depend on X_0, $X(t)$ is the true solution from equation (1) and $Y(t)$ defined as in (11).

Proof. See [KP95] Theorem 10.2.2 and Remark 10.2.3. □

Remark 2.3

The convergence presented above is just one of several possible convergence results. The Euler scheme also converges in other senses, but we choose this one to compare it to later results.

We see that the Euler scheme for SDEs is convergent of order $\frac{1}{2}$ which is not similar to the order of 1 as in the case of the Euler scheme for ODEs. Hence, we observe that by including stochasticity we lose rate of convergence. On another note, a convergence rate of $\frac{1}{2}$ is slow and we want a faster convergence. Therefore, we are next going to present the Milstein scheme for SDEs.

The Milstein scheme is an extension of the Euler scheme and when we consider the SDE from equation (1), the Milstein scheme is given by

$$\begin{aligned} Y_{n+1} &= Y_n + a(\tau_n, Y_n)\cdot\Delta + b(\tau_n, Y_n)\cdot(W_{\tau_{n+1}} - W_{\tau_n}) \\ &+ \frac{1}{2}b(\tau_n, Y_n)\cdot b'(\tau_n, Y_n)\cdot((\Delta W)^2 - \Delta), \\ Y_0 &= X_0. \end{aligned}$$

So in comparison to the Euler scheme we add the term $\frac{1}{2}bb'((\Delta W)^2 - \Delta)$.

Under certain conditions (see [KP95] Theorem 10.3.5), it is shown (see [KP95] Theorem 10.6.3) that for the Milstein approximation, it holds

$$\mathbb{E}\left(\sup_{0\le t\le T}|X(t) - Y(t)|\right) \le C\Delta, \tag{13}$$

where C depends on the starting value X_0, but not on $\Delta, Y(t)$ is our continuous approximation of the Milstein scheme and $X(t)$ the true solution.

We see that the Milstein scheme has a rate of convergence of 1 and is therefore converging faster than the Euler scheme with rate $\frac{1}{2}$.

So far we just looked at usual SDEs meaning that the functions a and b in equation (1) just depend on the present value of the process $X(t)$. This also means that the future state of our system is only determined by the present and not by past values. In many practical situations, this is not realistic, for example there may be stock prices which depend on the season of the year. Then if we just look at the current state we might not see this causality, but once we include past data it becomes obvious. Hence, we need to generalize our model. This is done by introducing stochastic functional differential equations (SFDEs) which allows exactly what we want: we can give more input (in our case past data) to our system. Following, we are going to take a look at the Euler scheme for SFDEs and its convergence.

We write the general SFDE as

$$dX(t) = f(X_t)dt + g(X_t)dW(t), \quad t \geq 0, \tag{14}$$

with initial data $X_0 = \xi \in L^p_{\mathcal{F}_0}([-\tau, 0]; \mathbb{R}^n)$. Here $\tau > 0$ is a fixed and given constant which determines how far we look into the past. In this case

$$f : \mathcal{C}([-\tau, 0]; \mathbb{R}^n) \to \mathbb{R}^n, \quad g : \mathcal{C}([-\tau, 0]; \mathbb{R}^n) \to \mathbb{R}^{n \times m},$$

$X(t) \in \mathbb{R}^n$ for each t, $W(t)$ is an $m-$dimensional Wiener process and

$$X_t = \{X(t + \theta) : -\tau \leq \theta \leq 0\} \in \mathcal{C}([-\tau, 0]; \mathbb{R}^n).$$

This is the general setup for SFDEs. However, in our applications we are only going to look at $1-$dimensional problems, so we can assume $m = 1$ throughout this thesis. The Euler scheme for SFDEs is more difficult than the one for regular SDEs although there are a lot of similarities. We use the same notations as [Mao03] (except that we still call our Wiener process W). Let $\Delta \in (0, 1)$ be our constant stepsize given by $\Delta = \tau/N$ for some integer

$N > \tau$ (similar to the SDE case N is the total amount of steps). We call our discrete Euler approximate solution $\bar{y}(k\Delta), k \geq -N$ and it is given by

$$\begin{aligned} \bar{y}(k\Delta) &= \xi(k\Delta) && -N \leq k \leq 0, \\ \bar{y}((k+1)\Delta) &= \bar{y}(k\Delta) + f(\bar{y}_{k\Delta})\Delta + g(\bar{y}_{k\Delta})\Delta W_k && k \geq 0, \end{aligned}$$

where $\bar{y}_{k\Delta} = \{\bar{y}_{k\Delta}(\theta) : -\tau \leq \theta \leq 0\}$ is a $\mathcal{C}([-\tau, 0]; \mathbb{R}^n)$-valued random variable about which we are going to talk after the comparison between this scheme and the one for the regular case. Here $\Delta W_k := W_{(k+1)\Delta} - W_{k\Delta}$ is the simulated increment of our Wiener process at time k. First, we observe large similarities to the regular case, since one time step is the previous time step plus something dependent on the step size and f plus something dependent on a Wiener process and g. This is exactly as in the previous case. Additionally, in this case we set our solution to the starting value for $-N \leq k \leq 0$. What makes this scheme more difficult is the term $\bar{y}_{k\Delta}$ and this is why we need to take a closer look at it.
$\bar{y}_{k\Delta}(\theta)$ is defined as follows:

$$\bar{y}_{k\Delta}(\theta) = \bar{y}((k+i)\Delta) + \frac{\theta - i\Delta}{\Delta}[\bar{y}((k+i+1)\Delta) - \bar{y}((k+i)\Delta)], \tag{15}$$

for $i\Delta \leq \theta \leq (i+1)\Delta, i = -N, -(N-1), ..., -1$. We see that $\bar{y}_{k\Delta}(\cdot)$ is the linear interpolation of $\bar{y}((k-N)\Delta), \bar{y}((k-N+1)\Delta, ..., \bar{y}(k\Delta)$. We can rewrite equation (15) as

$$\bar{y}_{k\Delta}(\theta) = \frac{\Delta - (\theta - i\Delta)}{\Delta}\bar{y}((k+i)\Delta) + \frac{\theta - i\Delta}{\Delta}\bar{y}((k+i+1)\Delta).$$

Hence, it holds

$$\begin{aligned} |\bar{y}_{k\Delta}(\theta)| &\leq \left|\frac{\Delta - (\theta - i\Delta)}{\Delta}\bar{y}((k+i)\Delta)\right| + \left|\frac{\theta - i\Delta}{\Delta}\bar{y}((k+i+1)\Delta)\right| \\ &\leq |\bar{y}((k+i)\Delta)| \vee |\bar{y}((k+i+1)\Delta)|. \end{aligned}$$

Therefore, we have

$$||\bar{y}_{k\Delta}|| = \max_{-N\leq i\leq 0} |\bar{y}((k+i)\Delta)| \text{ for all } k \geq 0.$$

Now, as for the Euler scheme for SDEs, we look at when this generalized Euler scheme converges and also in which sense. The following conditions will be needed.

Condition 2.4

Local Lipschitz condition:

For each $j \geq 1$, there is a right-continuous nondecreasing function $\mu_j : [-\tau, 0] \to \mathbb{R}_+$ such that

$$|f(\phi) - f(\psi)|^2 \vee |g(\phi) - g(\psi)|^2 \leq \int_{-\tau}^{0} |\phi(\theta) - \psi(\theta)|^2 d\mu_j(\theta),$$

for $\phi, \psi \in \mathcal{C}([-\tau, 0]; \mathbb{R}^n)$ with $||\phi|| \vee ||\psi|| \leq j$.

Condition 2.5

Linear growth condition:

There is a constant $K > 0$ such that

$$|f(\phi)|^2 \vee |g(\phi)|^2 \leq K(1 + ||\phi||^2)$$

for all $\phi \in \mathcal{C}([-\tau, 0]; \mathbb{R}^n)$ and $||\cdot||$ the supremum norm.

Condition 2.6

Initial condition:

$X_0 \in \mathcal{L}^p_{\mathcal{F}_0}([-\tau, 0]; \mathbb{R}^n)$ for some $p > 2$.

Remark 2.7

In our applications we are always going to have deterministic starting values meaning that

condition (2.6) is always fulfilled.

We are now able to formulate the convergence result for the Euler scheme for SFDEs.

Theorem 2.8

Under Conditions (2.4) - (2.6), the Euler scheme for SFDEs converges in the following sense:

$$\mathbb{E}\left(\sup_{0\leq t\leq T}|X(t)-Y(t)|^2\right)=\mathcal{O}(\Delta),\ \text{for all } T>0, \tag{16}$$

where $X(t)$ is the true solution from equation (14) and $Y(t)$ is defined as following: We set

$$\bar{y}_t := \sum_{n=0}^{N} Y_n \cdot \mathbb{1}_{[\tau_n,\tau_{n+1})}(t)$$

and afterwards we are able to define $Y(t)$ as

$$Y(t) := \begin{cases} \xi(t), & -\tau \leq t \leq 0, \\ \xi(0) + \int_0^t f(\bar{y}_s)ds + \int_0^t g(\bar{y}_s)dW(s), & t \geq 0. \end{cases}$$

Remark 2.9

We see that the Euler scheme for SFDEs converges in the same sense as the original Euler scheme for SDEs. Moreover, the conditions are very much alike, but instead of functions the conditions are extended to functionals.

3 Two mean SFDEs

In this chapter we consider a specific type of SFDEs and look at two examples. We are going to look at their Euler and their Milstein schemes. Furthermore, we examine the convergence and stability of these SFDEs and look at the existence of an invariant measure from a numerical point of view.

We consider stochastic functional differential equations of the form

$$dX(t) = f(X(t), \mathscr{T}(X)(t))dt + g(X(t), \mathscr{T}(X)(t))dW(t), \tag{17}$$

where f, g are appropriate functions, W is a Wiener process, $X(t) \in \mathbb{R}^n$, the mapping

$$\mathscr{T} : \mathcal{C}([0,\infty), \mathbb{R}^n) \ni X \to \mathscr{T}(X) \in \mathcal{C}([0,\infty), \mathbb{R}^n) \tag{18}$$

is measurable and $\mathscr{T}(X)(t)$ is progressive for each $t \geq 0$. For example, we can consider $\mathscr{T}(X)$ of the forms

(i) moving average: $\mathscr{T}(X)(t) := \frac{1}{\delta}\int_{(t-\delta)\vee 0}^{t} X(s)ds$ for some $\delta > 0$,

(ii) running average: $\mathscr{T}(X)(t) := \frac{1}{t}\int_0^t X(s)ds$ for $t > 0$ and $\mathscr{T}(X)(0) = X_0$,

(iii) running minimum or maximum: $\mathscr{T}(X)(t) := \inf_{s\in[0,t]} X(s)$ or
$\mathscr{T}(X)(t) := \sup_{s\in[0,t]} X(s)$.

Condition 3.1

Assume that the mapping $\mathscr{T}$ is Lipschitz with respect to the sup norm:

$$\|\mathscr{T}(X_1) - \mathscr{T}(X_2)\|_t \leq \kappa\|X_1 - X_2\|_t, \quad \forall X_1, X_2 \in \mathcal{C}([0,\infty), \mathbb{R}^n), \tag{19}$$

where κ is a positive constant, and $\|X\|_t := \sup_{0\leq s\leq t} |X(s)|$ is the sup norm on $\mathcal{C}([0,\infty), \mathbb{R}^n)$.

We can verify immediately that the moving and running average and running minimum or maximum operators are Lipschitz. Here we verify that the running average operator $\mathscr{T}(X)(t) = \frac{1}{t}\int_0^t X(s)ds$ is Lipschitz continuous with respect to the sup norm. Indeed, for any $X_1, X_2 \in \mathcal{C}([0,\infty),\mathbb{R})$, $t > 0$ and $0 < s \leq t$, we have

$$\begin{aligned}
|\mathscr{T}(X_1)(s) - \mathscr{T}(X_2)(s)| &= \left|\frac{1}{s}\int_0^s X_1(r)dr - \frac{1}{s}\int_0^s X_2(r)dr\right| \\
&\leq \frac{1}{s}\int_0^s |X_1(r) - X_2(r)|dr \\
&\leq \frac{1}{s}\int_0^s \sup_{u\in[0,t]} |X_1(u) - X_2(u)|dr \\
&= \sup_{u\in[0,t]} |X_1(u) - X_2(u)| \\
&= ||X_1 - X_2||_t.
\end{aligned}$$

Taking the supremum over $s \in [0,t]$ yields

$$\|\mathscr{T}(X_1) - \mathscr{T}(X_2)\|_t \leq \|X_1 - X_2\|_t,$$

showing that $\mathscr{T}$ is indeed Lipschitz.

Condition 3.2

Assume also that the functions f, g satisfy the following conditions:

$$|f(x,x')| + |g(x,x')| \leq K(1 + |x| + |x'|), \tag{20}$$

$$|f(x,x') - f(y,y')| + |g(x,x') - g(y,y')| \leq K(|x - x'| + |y - y'|), \tag{21}$$

for all $x, x', y, y' \in \mathbb{R}^n$.

We expect that when we suppose that Condition 3.1 and 3.2 hold, the SFDE (17) has a unique solution.

We now introduce the two SFDEs

$$dX(t) = (\bar{X}_t - X(t))dt + \sigma dW(t),\ t \in [0,T] \tag{22}$$

with $\sigma > 0$ and $\bar{X}_t = \frac{1}{t}\int_0^t X_s\, ds$ and

$$dX(t) = (\bar{X}_t - X(t))dt + \sigma X(t) dW(t),\ t \in [0,T]. \tag{23}$$

We observe that the SFDEs are very similar to each other, the difference is that while the first SFDE has a constant volatility, the second one has a scaling volatility.
We now present the Euler scheme for each of the two SFDEs. With the results from chapter 2 about the Euler scheme for SFDEs, we deduce that the Euler scheme for the first SFDE reads as

$$Y_{n+1} = Y_n + (\bar{Y}_n - Y_n)\cdot\Delta + \sigma\cdot\Delta W, \quad Y_0 = X_0 \tag{24}$$

where $\bar{Y}_n = \frac{1}{n+1}\sum_{i=0}^n Y_i$.
Similarly, the Euler scheme for the SFDE from equation (23) is given by

$$Y_{n+1} = Y_n + (\bar{Y}_n - Y_n)\cdot\Delta + \sigma\cdot Y_n\cdot\Delta W, \quad Y_0 = X_0. \tag{25}$$

We have seen that the Milstein scheme is an extension of the Euler scheme. Since we do not have any convergence results for the scheme in the case of SFDEs we are going to focus on the Euler scheme. However, for our two specific SFDEs we shall present the scheme.
In chapter 2 we have seen that the transition from the Euler scheme for SDEs to the Euler scheme for SFDEs is rather simple. For the Milstein scheme we see that the transition might get a little tricky, since we need the derivative of b (or in the SFDE case of g). However, this transition is only difficult if we use functionals for the stochastical part of the SFDE and in

this thesis we are only going to look at SFDEs where we only model the non-stochastic part, f, by a functional.

In the case of the SFDE from equation (22) the Milstein scheme is very easy, since $g \equiv \sigma$ and hence $g' = 0$. Therefore, for this SFDE the Milstein scheme is the same as the Euler scheme.

The Milstein scheme for the SFDE from equation (23) is not the same as the Euler scheme, since it is given by

$$Y_{n+1} = Y_n + (\bar{Y}_n - Y_n) \cdot \Delta + \sigma \cdot Y_n \cdot \Delta W + \frac{1}{2}\sigma^2 \cdot Y_n \cdot ((\Delta W)^2 - \Delta), Y_0 = X_0. \tag{26}$$

There is a very famous family of SDEs called Ornstein-Uhlenbeck processes (OU processes) which is closely related to our two SFDEs above. It is given by

$$dX(t) = \theta(\mu - X(t))dt + \sigma dW(t), \quad X_0 = a, \tag{27}$$

where μ is called the mean-reversion level and it is the value around which the SDE evolves, θ is called the mean-reversion speed and indicates how fast and strong the SDE evolves around μ and σ determines the impact of randomness. The OU process has an explicit solution which is given by

$$X(t) = \mu + (a - \mu)\exp(-\theta t) + \sigma \int_0^t \exp(-\theta(t - s))dW_s.$$

We now show how we come up with this explicit solution. We rewrite (27) as

$$dX(t) + \theta X(t)dt = \theta\mu dt + \sigma dW(t).$$

Multiply both sides of the equation by $e^{\theta t}$ to obtain

$$e^{\theta t}dX(t) + \theta e^{\theta t}X(t)dt = \theta\mu e^{\theta t}dt + \sigma e^{\theta t}dW(t).$$

By the integration by parts formula, we have

$$d(e^{\theta t}X(t)) = e^{\theta t}(dX(t) + \theta X(t)dt).$$

Thus

$$d(e^{\theta t}X(t)) = \theta\mu e^{\theta t}dt + \sigma e^{\theta t}dW(t).$$

Now integrating both sides of the equation gives

$$e^{\theta t}X(t) - X(0) = \int_0^t \theta\mu e^{\theta s}ds + \int_0^t \sigma e^{\theta s}dW(s) = \mu(e^{\theta t} - 1) + \int_0^t \sigma e^{\theta s}dW(s).$$

Then it follows that

$$X(t) = \mu + (X(0) - \mu)e^{-\theta t} + \int_0^t \sigma e^{\theta(s-t)}dW(s).$$

The stochastic integral has mean zero and hence $\mathbb{E}[X(t)] = \mu + (X(0) - \mu)e^{-\theta t}$. In particular, this shows that μ is the long-term mean of the Ornstein–Uhlenbeck process.

Next we use the method in [KT81], Section 15.5 to determine the stationary distribution of (27). Let

$$s(x) := \exp\left\{ -\int_0^x \frac{2\theta(\mu - y)}{\sigma^2}dy \right\} = \exp\left\{ -\frac{2\theta\mu}{\sigma^2}x + \frac{\theta}{\sigma^2}x^2 \right\}, \quad x \in \mathbb{R},$$

and

$$m(x) := \frac{1}{\sigma^2 s(x)} = \frac{1}{\sigma^2}\exp\left\{ \frac{2\theta\mu}{\sigma^2}x - \frac{\theta}{\sigma^2}x^2 \right\}, \quad x \in \mathbb{R}.$$

Note that $m(x) > 0$ and $\int_{-\infty}^{\infty} m(x)dx < \infty$, it follows that the stationary density of (27) is given by $Cm(x)$, where C is the normalizing constant so that $\int_{-\infty}^{\infty} Cm(x)dx = 1$.

There is a SDE which is closely related to the SFDE from equation (23) given by

$$dX(t) = \theta(\mu - X(t))dt + \sigma X(t)dW(t), \quad X_0 = a, \tag{28}$$

with the same explanations as for the OU process. This SDE does not have a constant volatility, but a scaling volatility as the SFDE from (23). Additionally, also this SDE has an explicit solutions which is given by

$$X(t) = \frac{a}{M(t)} + \theta\mu \int_0^t \frac{M(s)}{M(t)} ds,$$

where $M(t) = \exp((\theta + \frac{\sigma^2}{2})t - \sigma W(t))$. Again, we are going to show how we derive the explicit solutions in this case. To derive the solution of (28), we consider the auxilliary process $M(t) := e^{Y(t)}$, where $Y(t) := (\theta + \frac{1}{2}\sigma^2)t - \sigma W(t)$. Consider the function $f(x) = e^x$. We have $f'(x) = f''(x) = f(x)$. Thus by Itô's formula, we have

$$dM(t) = df(Y(t)) = f'(Y(t))dY(t) + \frac{1}{2}f''(Y(t))\sigma^2 dt = M(t)(\theta + \sigma^2)dt - \sigma M(t)dW(t).$$

Then using the integration by parts formula, we obtain

$$\begin{aligned} d(X(t)M(t)) &= X(t)dM(t) + M(t)dX(t) + d[M, X]_t \\ &= X(t)M(t)(\theta + \sigma^2)dt - X(t)\sigma M(t)dW(t) \\ &\quad + M(t)\theta(\mu - X(t))dt + M(t)\sigma X(t)dW(t) - \sigma^2 X(t)M(t)dt \\ &= \theta\mu M(t)dt. \end{aligned}$$

Then it follows that

$$M(t)X(t) - M(0)X(0) = \int_0^t \theta\mu M(s)ds,$$

and hence

$$\begin{aligned} X(t) &= \frac{X(0)}{M(t)} + \theta\mu \int_0^t \frac{M(s)}{M(t)} ds \\ &= X(0)e^{-(\theta+\frac{1}{2}\sigma^2)t+\sigma W(t)} + \theta\mu \int_0^t e^{-(\theta+\frac{1}{2}\sigma^2)(t-s)+\sigma(W(t)-W(s))} ds. \end{aligned}$$

Since $W(t) \sim \mathcal{N}(0,t)$ and $W(t) - W(s) \sim \mathcal{N}(0, t-s)$, one can use the moment generating function of a normal random variable to see that

$$\mathbb{E}[e^{-\frac{1}{2}\sigma^2 t+\sigma W(t)}] = \mathbb{E}[e^{-\frac{1}{2}\sigma^2(t-s)+\sigma(W(t)-W(s))}] = 1.$$

Then we have

$$\mathbb{E}[X(t)] = X(0)e^{-\theta t} + \theta\mu \int_0^t e^{-\theta(t-s)} ds = \mu + (X(0) - \theta\mu)e^{-\theta t}.$$

Similar to calculations for the process (27), the process (28) also has a stationary density function given by

$$Cm(x) = \frac{C}{\sigma^2} \exp\left\{ -\frac{2\theta\mu}{\sigma^2 x} \right\} x^{-2-\frac{2\theta}{\sigma^2}}, \quad x > 0,$$

where $C > 0$ is the normalizing constant so that $\int_0^\infty Cm(x)dx = 1$. Note that this is the inverse gamma distribution.

When we compare the two regular SDEs to our two SFDEs we see that we implicitly chose $\theta = 1$. We also observe that while our SDEs evolve around μ, our SFDEs evolve around their mean. Additionally, the first SFDE (22) also has a constant volatility just like the OU process, while on the other hand our second SFDE (23) has scaling volatility like the SDE from equation (28). Hence, we observe that the two SDEs are closely related to our two SFDEs.

What we take a look of now is, how we can estimate the parameter σ in equation (22).

Possibly, there are several ways for this estimation, e.g. an MLE-estimator, but we choose an intuitive one. When we solve equation (24) for σ, we deduce

$$\sigma = \frac{Y_{n+1} - Y_n - (\bar{Y}_n - Y_n)\Delta}{\Delta W}. \tag{29}$$

Since $\sigma > 0$ has to hold we can take the absolute value of the right hand side. The larger problem is that we would divide by $|\Delta W|$ which is random and would not make any sense for an estimation. To solve this problem we take the expected value of $|\Delta W|$. Hence, we can define

$$\sigma_n := \frac{|Y_{n+1} - Y_n - (\bar{Y}_n - Y_n)\Delta|}{\mathbb{E}|\Delta W|} = \frac{|Y_{n+1} - Y_n - (\bar{Y}_n - Y_n)\Delta|}{\sqrt{\frac{2\Delta}{\pi}}}, \tag{30}$$

since it holds

$$\mathbb{E}(|\Delta W|) = \int_{-\infty}^{\infty} |x| \frac{1}{\sqrt{2\pi\Delta}} \exp\left(-\frac{x^2}{2\Delta}\right) dx = \sqrt{\frac{2\Delta}{\pi}}.$$

Afterwards, we use Monte Carlo simulation to estimate σ as

$$\hat{\sigma} = \frac{1}{N} \sum_{n=0}^{N-1} \sigma_n. \tag{31}$$

We are testing this procedure by simulating paths with the Euler scheme for known σ and afterwards control whether our estimated $\hat{\sigma}$ is close to the true σ. These numerical tests show that this is a good estimate for σ, when we are given data $Y_0, Y_1, ...Y_N$.

We now want to estimate σ for the SFDE from equation (23). The estimation of σ is very similar to what we did in equations (30) and (31).

Hence, we define

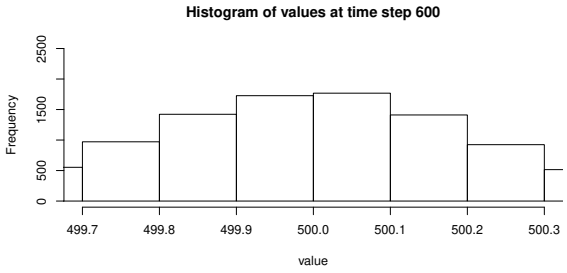
$$\sigma_n := \frac{|Y_{n+1} - Y_n - (\bar{Y}_n - Y_n)\Delta|}{Y_n \cdot \mathbb{E}|\Delta W|} = \frac{|Y_{n+1} - Y_n - (\bar{Y}_n - Y_n)\Delta|}{Y_n \cdot \sqrt{\frac{2\Delta}{\pi}}} \tag{32}$$

and as before estimate σ as the mean over all σ_n. Also in this case, our numerical tests which we used also in the previous application show that this is a good and reasonable estimation for σ.

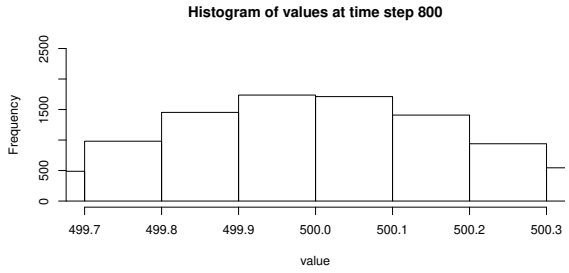
So far in this chapter we presented the Euler scheme and the Milstein scheme for two similar SFDEs. We now want to take a numerical look at the distribution of those SFDEs. Especially, we are interested in the fact whether these distributions are stable, stationary and converge at some point and thereby an invariant measure exists.

We have seen that the two SDEs from (27) and (28) possess a stationary distribution and hence an invariant measure exists. Since our SFDEs are very similar our first guess is that our distributions also have stationary distributions.

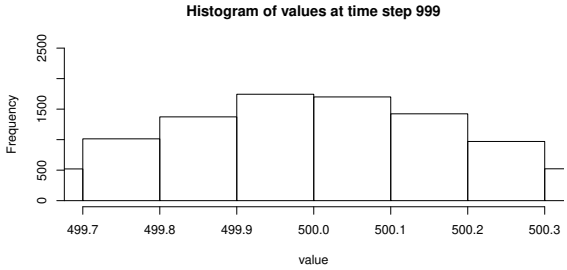
We are not going to give a proof of the existence of an invariant measure, but we take a look at it from a numerical point of view. This is done by simulating paths of each SFDE and then calculating the estimated probability density function (epdf) for different time steps. Then we compare the epdf of different time steps to each other and if the epdf does not change much this indicates that the true distribution converges to a stationary distribution. We simulate 10000 paths for each SFDE to determine the corresponding epdf.



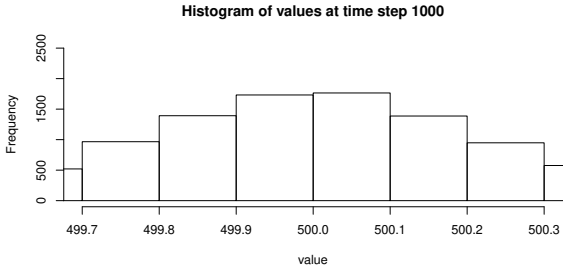
(a) Histogram corresponding to the epdf at timestep 600



(b) Histogram corresponding to the epdf at timestep 800



(c) Histogram corresponding to the epdf at timestep 999

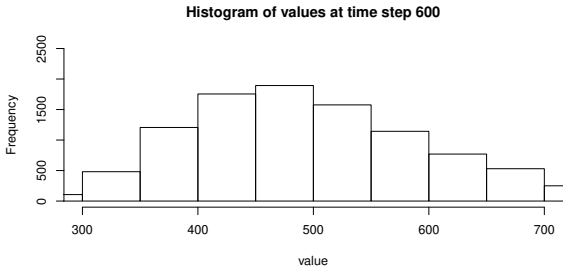


(d) Histogram corresponding to the epdf at timestep 1000

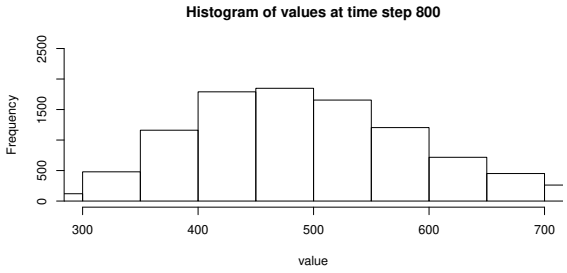
Figure 1: Histograms from different time steps 1

We consider the SFDE from equation (22) with parameters $\sigma = 0.2$ and $X_0 = 500$. Additionally, we look at a time period of 100 years discretized in 1000 time steps. We use an equidistant stepwidth which is then given by $100/1000 = 0.1$. We are going to approximate the pdf's X_{600} (df of the SFDE at time step 600), X_{800} (df of the SFDE at time step 800), X_{999} (df of the SFDE at time step 999) and X_{1000} (df of the SFDE at time step 1000 which is also the last step). The corresponding R-code can be found in the appendix (program 1).

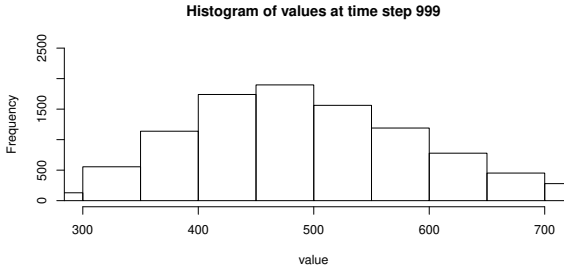
The plots from figure (1) show that the histograms and therefore the epdfs and the distribution functions do not change much from one time step to the other. Hence, we conclude that the distribution function of the SFDE from equation (22) is stationary and therefore the SFDE has an invariant measure.



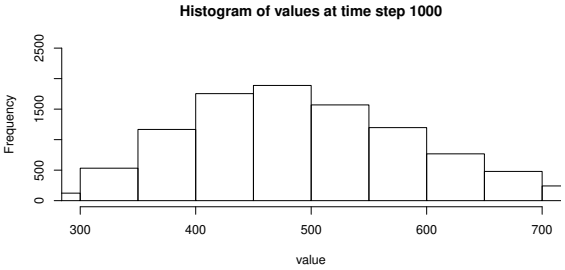
(a) Histogram corresponding to the epdf at timestep 600



(b) Histogram corresponding to the epdf at timestep 800



(c) Histogram corresponding to the epdf at timestep 999



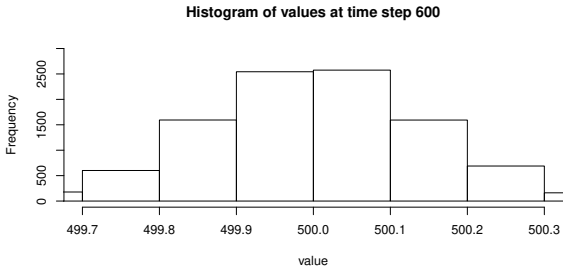
(d) Histogram corresponding to the epdf at timestep 1000

Figure 2: Histograms from different time steps 2

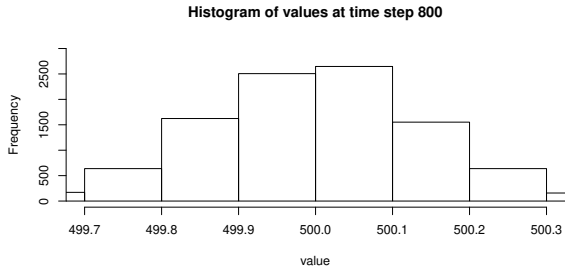
We now do the same thing for the SFDE from equation (23) with the same parameters as for the other SFDE ($\sigma = 0.2, X_0 = 500, 100$ years, 1000 time steps). The corresponding R-code can be found in the appendix (program 2).

These plots from figure (2) show that there is much larger deviation around the starting value compared to the previous SFDE. However, we see that the histograms do not differ much from each other which indicates, that again, the distribution is stationary and an invariant measure exists.

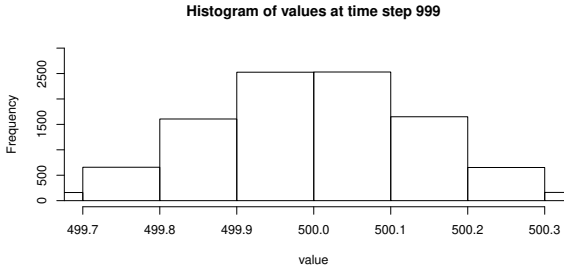
To compare results, we also show the corresponding histograms for the SDEs (27) and (28). We start with the OU process. To have the best possible comparison we choose the same parameters as before and additionally we choose $\mu = 500$, since the SFDE from (22) with our parameters evolves around 500 and we want the same for OU process. (same program



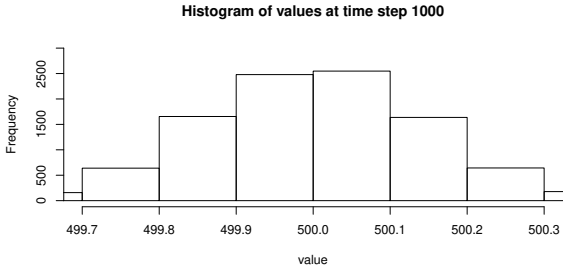
(a) Histogram corresponding to the epdf at timestep 600



(b) Histogram corresponding to the epdf at timestep 800



(c) Histogram corresponding to the epdf at timestep 999

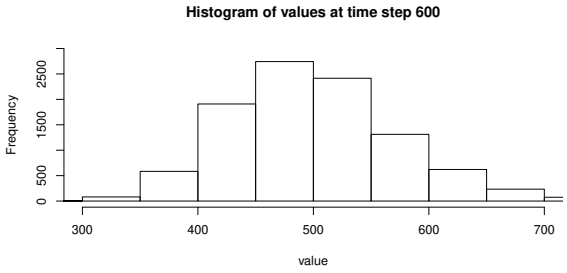


(d) Histogram corresponding to the epdf at timestep 1000

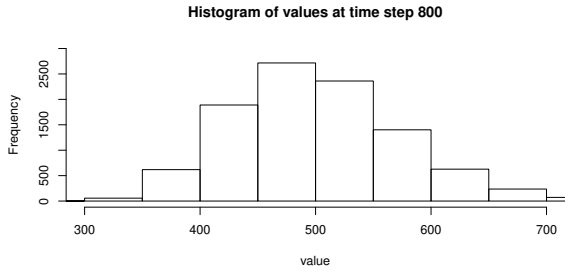
Figure 3: Histograms from different time steps 3

as program 1 from the appendix, instead of "mean(process)" we write "500")

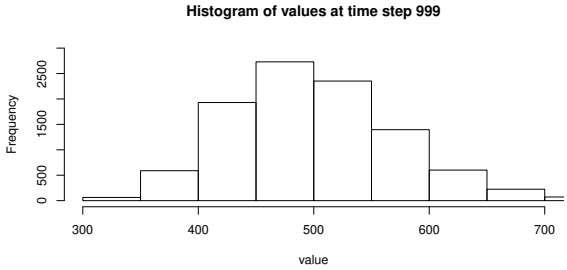
Figure (3) looks very similar to figure (1). They both evolve around 500 very closely and we again see that the distribution does not change much from one step to the other which makes sense since we have seen that we have a stationary distribution.



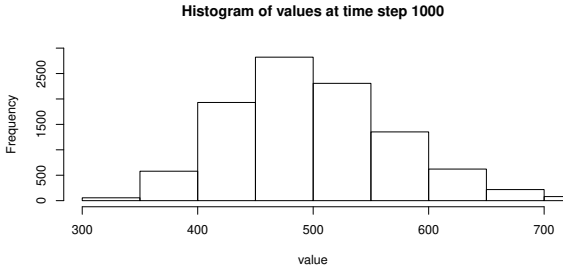
(a) Histogram corresponding to the epdf at timestep 600



(b) Histogram corresponding to the epdf at timestep 800



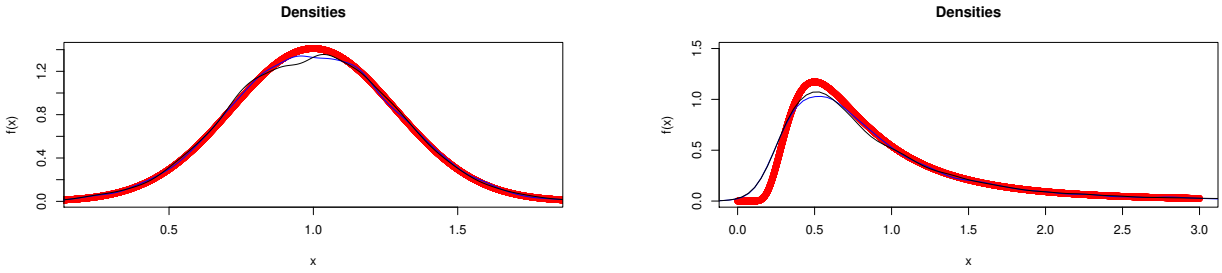
(c) Histogram corresponding to the epdf at timestep 999



(d) Histogram corresponding to the epdf at timestep 1000

Figure 4: Histograms from different time steps 4

Next, we present the histograms for the SDE (28). We take the same parameters as in the OU process case. (same program as program 2 from the appendix, instead of "mean(process)" we write "500"). We observe that the histograms from figure (4) are very much alike to the histograms from figure (2). They both have a much larger deviation around 500 when compared to figures (1) and (3). As before, we see that the distribution does not change much from one time step to the other showing that this distribution is indeed stationary.



(a) Densities OU process

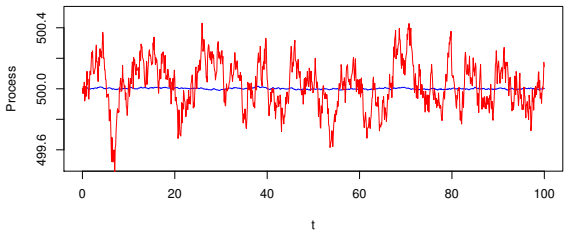
(b) Densities for SDE (28)

Figure 5: Densities for SDEs

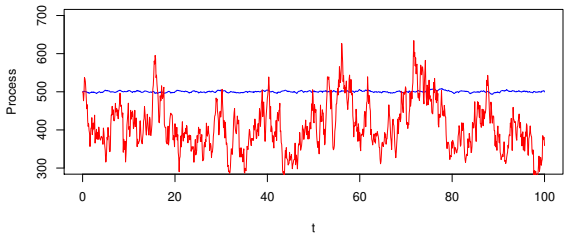
Additionally to the previous plots, we want to compare the theoretical stationary densities from the SDEs (27) and (28) to the densities we observe when computing paths with the Euler scheme. For simplification, we take different parameters than for the histograms. For the constants from the OU process we take $\theta = 1, \mu = 1, \sigma = 0.4, X_0 = 500$ and for the SDE from equation (28) we take $\theta = 1, \mu = 1, \sigma = 1, X_0 = 500$. In both cases, we have $N = 1000, T = 100$ and simulate 10001 paths (exemplary R-code, see program 3).

The thin blue and black lines show approximated densities from two consecutive time steps (here time steps 999 and 1000) and the large red line shows the theoretical density.

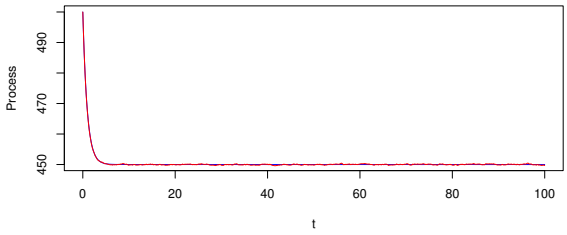
We observe that in both cases the theoretical density is close to our approximated densities. To get even closer to the theoretical density, we could consider the Milstein scheme or raise N or T.



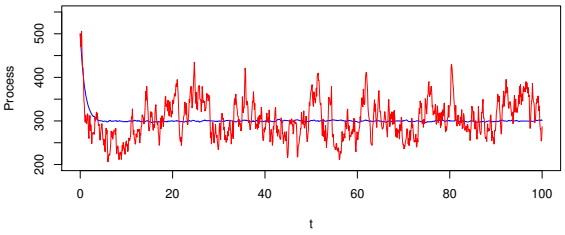
(a) Paths for SFDE from (22)



(b) Paths for SFDE from (23)



(c) Paths for OU process



(d) Paths for SDE from (28)

Figure 6: Sample and average paths

Lastly, to get a feeling for how our SDEs and SFDEs behave, we show plots of the average path (MC path with 1000 simulations) and one specific path for each of the four models. We always use the Euler schemes to simulate paths (exemplary R-code, see program 4) The average path is always going to be blue and the specific path red. We use $N = 1001, T = 100, \sigma = 0.2, \theta = 1$ and a starting value of 500. Additionally, for the OU process we use $\mu = 450$ and for the SDE from (28) we take $\mu = 300$.

4 Four models for the stock market

In this chapter we consider four models based on SDEs and SFDEs and look at how each of these models fit real data from the stock market.

4.1 Model 1

First, we are going to introduce the four models. The first models are based on the famous Black-Scholes SDE:

$$dX(t) = \mu X(t)dt + \sigma X(t)dW(t), \quad X_0 = X_{t_0}, \tag{33}$$

with $\mu \in \mathbb{R}$ and $\sigma > 0$. The Euler scheme of the Black-Scholes SDE is given by:

$$Y_{n+1} = Y_n + \mu \cdot Y_n \cdot \Delta + \sigma \cdot Y_n \cdot \Delta W, \quad Y_0 = X_0. \tag{34}$$

The first models are based on the same SDE, but we are going to use different estimators for μ and σ.

The first estimation is based on a numerical scheme for estimating μ and σ, very similar to how we estimated σ in chapter 2. We know that $\mathbb{E}(\Delta W) = 0$ holds. Hence, in expectation we can omit $\sigma \cdot Y_n \cdot \Delta W$ in equation (34) and afterwards solve for μ. We deduce

$$\mu = \frac{Y_{n+1} - Y_n}{Y_n \cdot \Delta}.$$

We now do this for every time step n which gives sense to the definitions

$$\mu_n := \frac{Y_{n+1} - Y_n}{Y_n \cdot \Delta} \text{ as well as } \hat{\mu} := \frac{1}{N} \sum_{n=0}^{N-1} \mu_n.$$

To estimate σ in this case, we proceed as in chapter 3 meaning that we take the absolute value and solve for σ. Since σ now depends on μ and we usually do not know the true value

of μ we replace μ by $\hat{\mu}$. Concluding, we set

$$\sigma_n := \frac{|Y_{n+1} - Y_n - \hat{\mu} Y_n \Delta|}{|Y_n| \, \mathbb{E}(|\Delta W|)} = \frac{|Y_{n+1} - Y_n - \hat{\mu} Y_n \Delta|}{|Y_n| \sqrt{\frac{2\Delta}{\pi}}},$$

with our knowledge about $\mathbb{E}(|\Delta W|)$. As before, we define

$$\hat{\sigma} := \frac{1}{N} \sum_{n=0}^{N-1} \sigma_n.$$

These estimators for μ and σ are specific for the Euler scheme.

In the plots we refer to this model as "Black-Scholes with Euler".

4.2 Model 2

Following, we consider a theoretical approach to estimate μ and σ in the Black-Scholes SDE. It is known that in the Black-Scholes model, it holds

$$\ln\left(\frac{Y_{n+1}}{Y_n}\right) \sim \mathcal{N}(\mu - \frac{\sigma^2}{2}\Delta, \sigma^2 \Delta)$$

where Y_n is given as in equation (34). This leads to the following procedure for estimating μ and σ based on the MLE for the lognormal distribution. We set

$$X_n := \ln\left(\frac{Y_{n+1}}{Y_n}\right), \bar{X} := \text{mean}(X_0, X_1, ..., X_{N-1}) \text{ and } \text{std}(X) := \sqrt{\text{Var}(X_0, X_1, ..., X_{N-1})}.$$

We now have to annualize these values. Therefore, we define

$$\bar{X}_a := T \cdot \bar{X} \text{ and } \text{std}(X)_a := \sqrt{T} \cdot \text{std}(X).$$

The estimators for μ and σ are now given by

$$\hat{\mu} := \bar{X}_a + \frac{(\text{std}(X)_a)^2}{2} \text{ and } \hat{\sigma} := \text{std}(X)_a.$$

This concludes the parameter estimation process of our two SDEs related to the Black-Scholes equation.

In the plots we refer to this model as "Black-Scholes theoretical".

4.3 Model 3

The third model what we are going to fit to real data is going to be the SFDE from equation (23). We already know from chapter 3 how the parameter estimation works for this process.

In the plots we refer to this model as "MeanSDE with Euler".

4.4 Model 4

The fourth and final model we introduce is a mix between the Black-Scholes SDE and the SFDE from equation (23). It has a dependence on the mean of the process as well as a dependence on a constant growing factor like in the Black-Scholes model. The SFDE is given by

$$dX(t) = (\bar{X}_t - X(t) + \mu X(t))dt + \sigma\ X(t)dW(t). \tag{35}$$

We deduce the corresponding Euler scheme as

$$Y_{n+1} = Y_n + (\bar{Y} - Y_n + \mu\ Y_n)\ \Delta + \sigma\ Y_n \Delta W. \tag{36}$$

Similarly to how we performed the parameter estimation for the first model, we estimate the parameter in this case. Therefore, our estimators for μ and σ for this model are given by

$$\mu_n := \frac{Y_{n+1} - Y_n - (\bar{Y} - Y_n)\Delta}{\Delta\, Y_n}, \quad \hat{\mu} := \frac{1}{N}\sum_{n=0}^{N-1} \mu_n$$

and

$$\sigma_n := \frac{|Y_{n+1} - Y_n - \hat{\mu} Y_n \Delta|}{|Y_n|\sqrt{\frac{2\Delta}{\pi}}}, \quad \hat{\sigma} := \frac{1}{N}\sum_{n=0}^{N-1} \sigma_n.$$

In the plots we refer to this model as "MeanSDE2 with Euler".

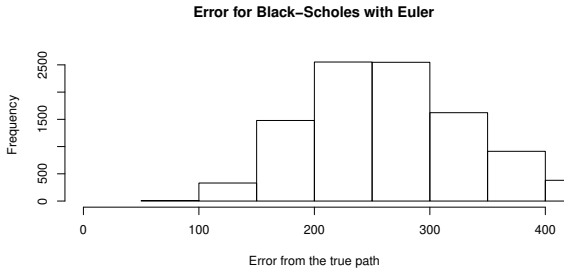
4.5 Data Analysis

We now have four models for the stock market and we know how to estimate the parameters for each one of these models. Now, we are going to apply this theory to real data observed on the American stock market. Therefore, we get the stock prices of companies (we always take the closing price for which the stock was traded on that day) and estimate our parameters for each model with these data. Afterwards, we simulate 10000 paths of each model with the estimated parameters. We somehow have to determine which models are good and which ones are bad. For each of our simulated paths, we therefore consider the supremum-metric

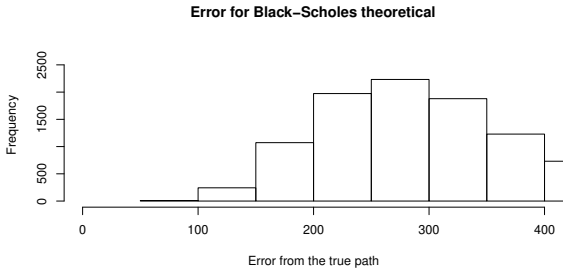
$$\sup_{0 \le n \le N} (|X_n - Y_n|),$$

where $X_n (= X(n))$ are the true and observed data, Y_n are the values from our simulated paths, N is the index of our last value (e.g. X_N is the last observed data) and n our time index (e.g. X_5 is the observed data of day 5). We calculate the supremum-metric for each path and each model. Afterwards, we plot histograms showing how often a certain error occured. If there is a high frequency of small errors and a low frequency of large errors, we

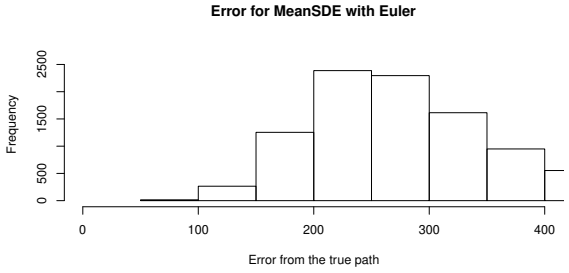
conclude that this model is a good model, otherwise it is a bad model. Additionally, we calculate the average error of each model to its respective paths and it is calculated as the average over all the error data in one histogram. We consider some different stocks, but we always look at data of one month as well as data of six months for the same stock to see how well each model does for different lengths of time. We assume that one year has 252 trading days. This information is important in that it determines our stepwidth Δ. Moreover, we consider an equidistant partition, since our data are always exactly one day apart from each other. Hence, Δ is constant and the same for each time step. On a short note we should say that we got all our data from the official NASDAQ website (www.nasdaq.com). The corresponding R-code to these simulations can be found in the appendix (program 5).



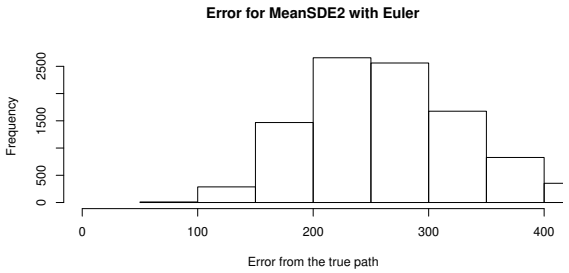
(a) Histogram for Model 1, meanerror = 269.02



(b) Histogram for Model 2, meanerror = 298.45



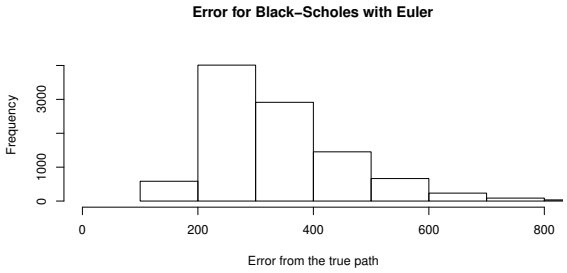
(c) Histogram for Model 3, meanerror = 289.58



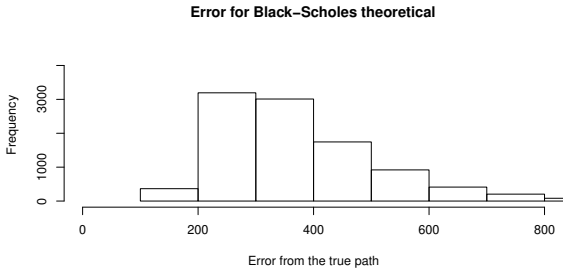
(d) Histogram for Model 4, average error = 268.15

Figure 7: Error histograms of our four models considering the Amazon stock prices from 01/24/2020 to 02/24/2020

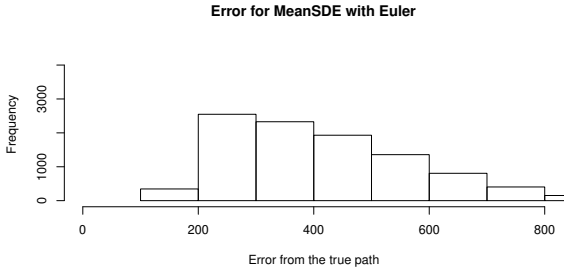
When we evaluate the four histograms for Amazon stock prices for one month (figure (7)), we observe that our last model is the best one closely followed by our first model. The second and the third model do not compare very well to the other two.



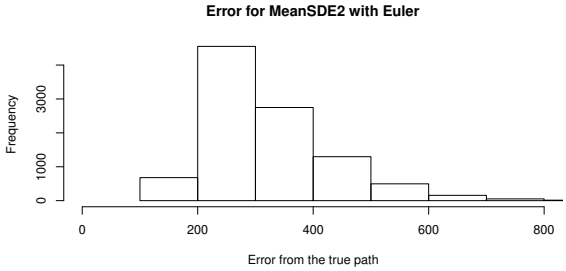
(a) Histogram for Model 1, average error = 338.97



(b) Histogram for Model 2, average error = 375.79



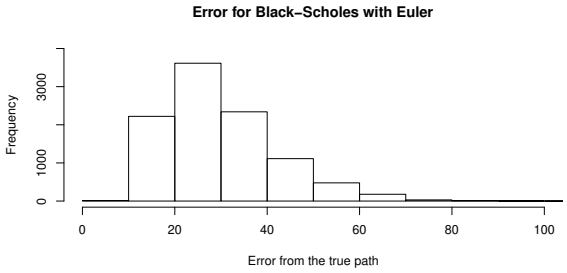
(c) Histogram for Model 3, average error = 420.17



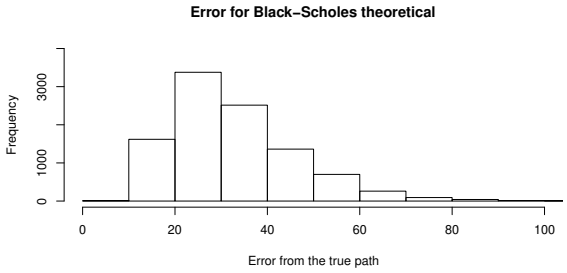
(d) Histogram for Model 4, average error = 320.78

Figure 8: Error histograms of our four models considering the Amazon stock prices from 08/26/2019 to 02/24/2020

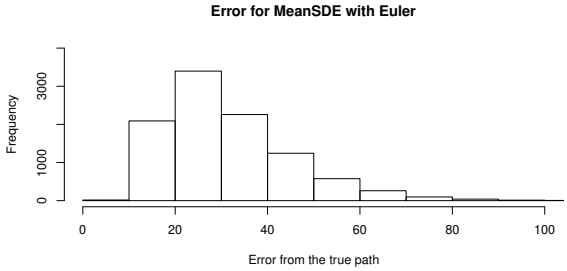
As for the one month stock prices we see that in the six month case (figure (8)) our last model performs the best, followed by our first model. The second and third model behave much worse.



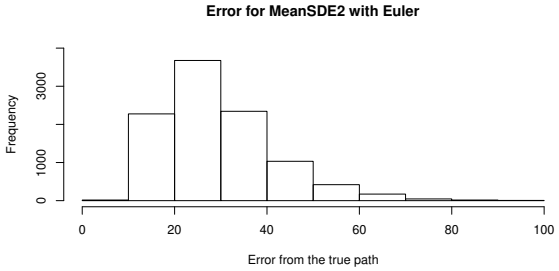
(a) Histogram for Model 1, average error = 29.68



(b) Histogram for Model 2, average error = 32.68



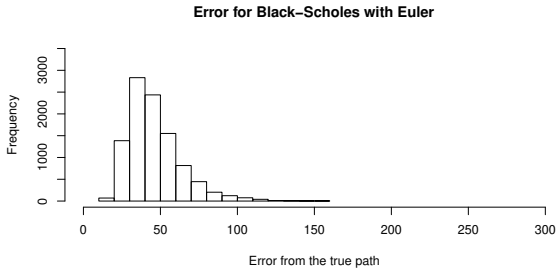
(c) Histogram for Model 3, average error = 31.31



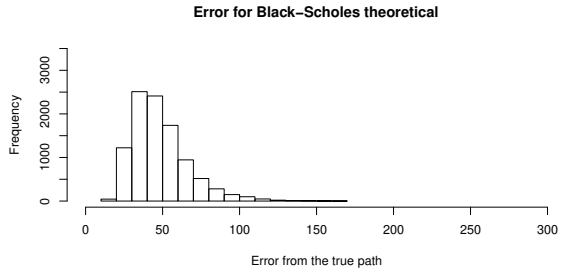
(d) Histogram for Model 4, average error = 29.46

Figure 9: Error histograms of our four models considering the Apple stock prices from 01/16/2020 to 02/14/2020

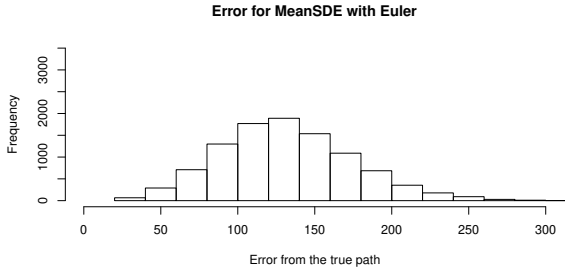
Figure (9) shows the Apple stock prices for one month and we observe that all of our models perform very similarly.



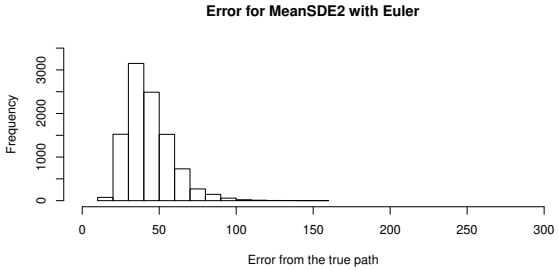
(a) Histogram for Model 1, average error = 46.08



(b) Histogram for Model 2, average error = 48.20



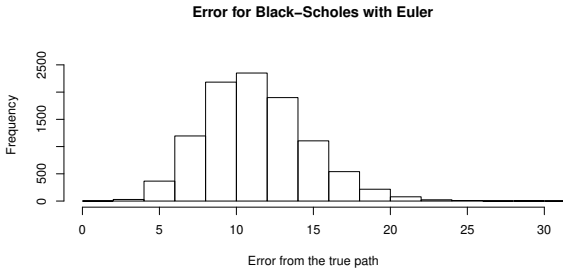
(c) Histogram for Model 3, average error = 131.90



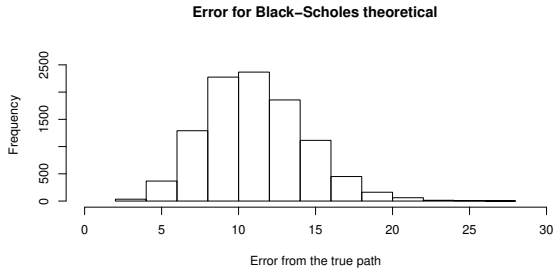
(d) Histogram for Model 4, average error = 43.50

Figure 10: Error histograms of our four models considering the Apple stock prices from 08/16/2019 to 02/14/2020

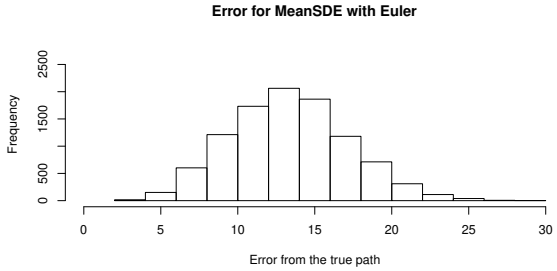
Figure (10) shows that all models perform similarly except the third model which behaves much worse than the other ones.



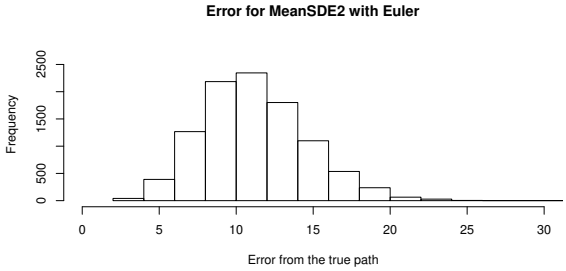
(a) Histogram for Model 1, average error = 11.30



(b) Histogram for Model 2, average error = 11.08



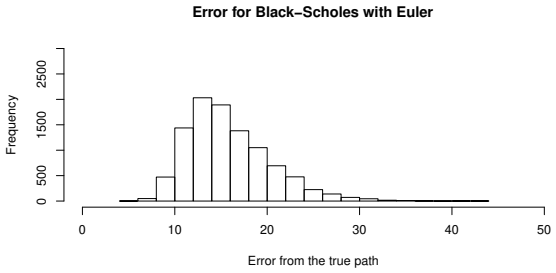
(c) Histogram for Model 3, average error = 13.35



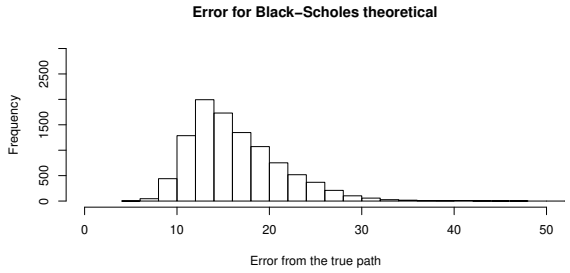
(d) Histogram for Model 4, average error = 11.20

Figure 11: Error histograms of our four models considering the Walmart stock prices from 02/03/2020 to 02/28/2020

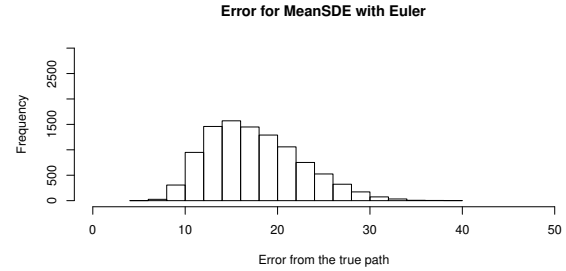
When we look at the one month data from Walmart (figure (11)) we see that our third model performs the worst while the other three models are very close to each other.



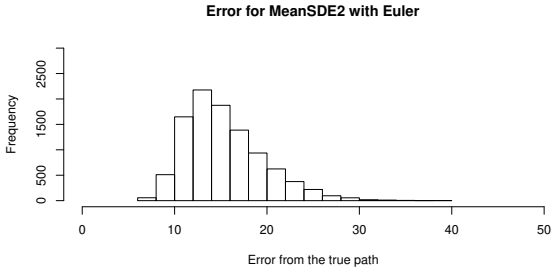
(a) Histogram for Model 1, average error = 15.82



(b) Histogram for Model 2, average error = 16.37



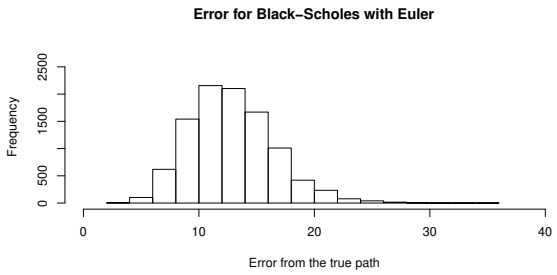
(c) Histogram for Model 3, average error = 17.51



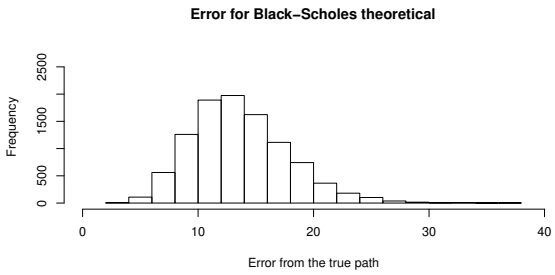
(d) Histogram for Model 4, average error = 15.35

Figure 12: Error histograms of our four models considering the Walmart stock prices from 09/03/2019 to 02/28/2020

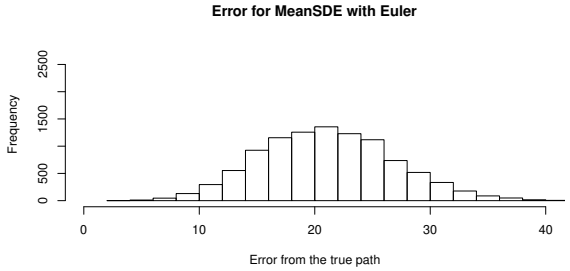
Similar to the previous plots, we observe in figure (12) that our fourth model performs the best, closely followed by our first model, afterwards our second model and last our third model.



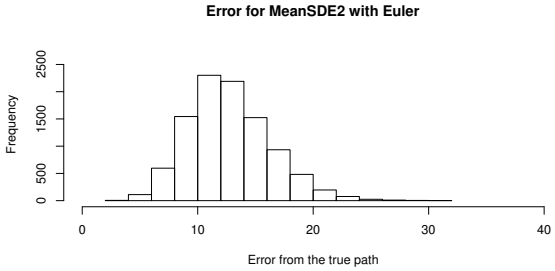
(a) Histogram for Model 1, average error = 12.83



(b) Histogram for Model 2, average error = 13.61



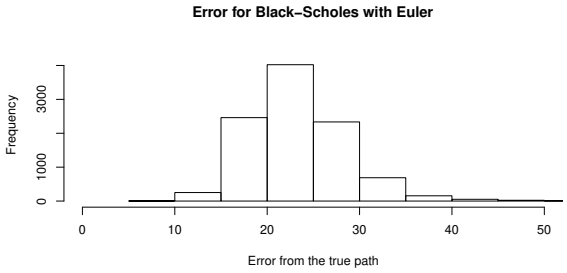
(c) Histogram for Model 3, average error = 21.10



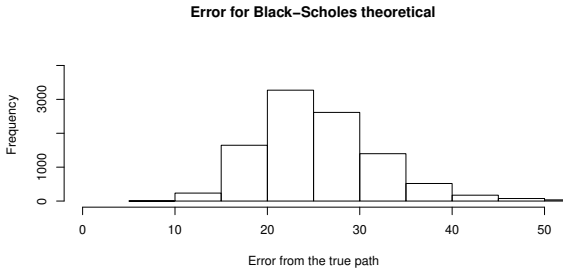
(d) Histogram for Model 4, average error = 12.72

Figure 13: Error histograms of our four models considering the Johnson & Johnson stock prices from 02/03/2020 to 02/28/2020

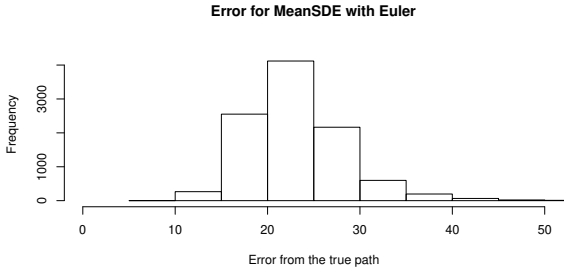
When we observe the different histograms from figure (13) which show how our models compare to each other for the one month stock price from Johnson & Johnson, we see that our first and our fourth model perform the best, closely followed by our second model while our third model performs really bad.



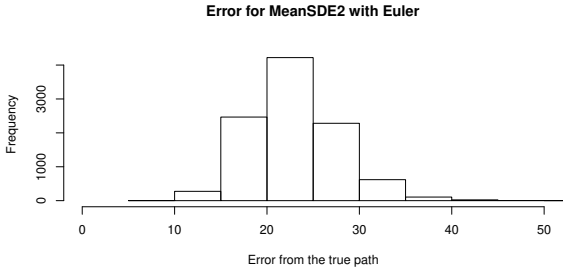
(a) Histogram for Model 1, average error = 23.31



(b) Histogram for Model 2, average error = 25.69



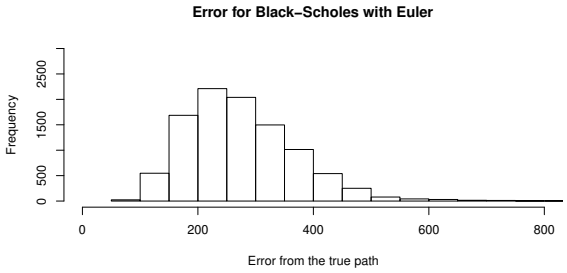
(c) Histogram for Model 3, average error = 23.17



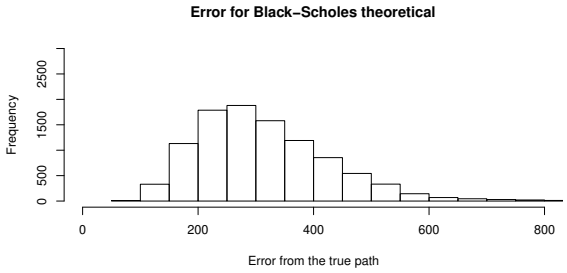
(d) Histogram for Model 4, average error = 22.97

Figure 14: Error histograms of our four models considering the Johnson & Johnson stock prices from 09/03/2019 to 02/28/2020

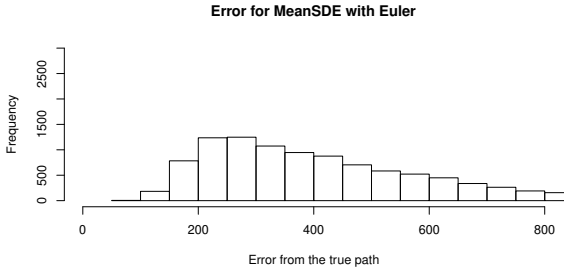
The six month stock prices from Johnson & Johnson (figure (14)) show that again our fourth model performs the best, directly followed by our third and first model. We see that for this data our second model performs the worst.



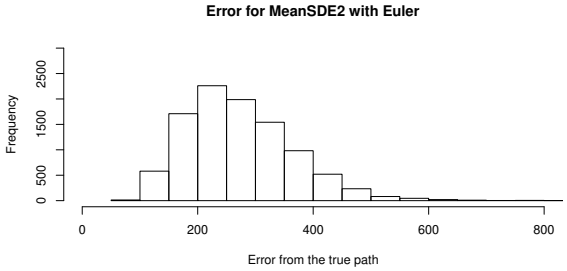
(a) Histogram for Model 1, average error = 275.04



(b) Histogram for Model 2, average error = 315.73



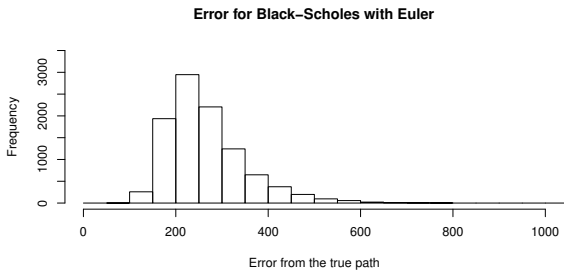
(c) Histogram for Model 3, average error = 422.22



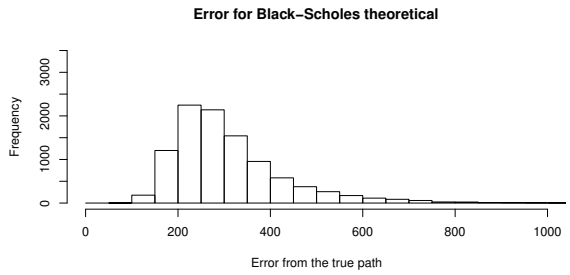
(d) Histogram for Model 4, average error = 273.14

Figure 15: Error histograms of our four models considering the Tesla stock prices from 01/24/2020 to 02/21/2020

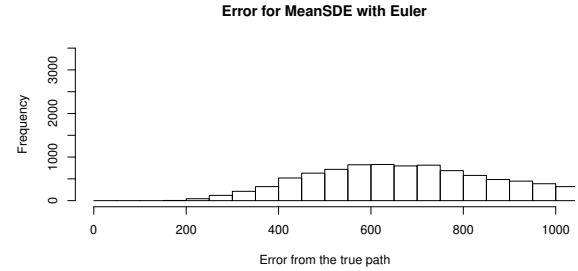
In figure (15) which shows the error plots for the one month Tesla stock price, we observe the same things as before. Our fourth and our first model perform the best, followed by our second model and last our third model.



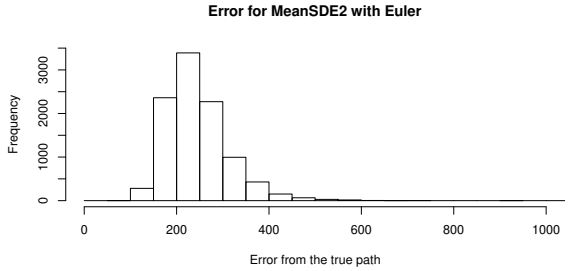
(a) Histogram for Model 1, average error = 264.90



(b) Histogram for Model 2, average error = 309.28



(c) Histogram for Model 3, average error = 739.24



(d) Histogram for Model 4, average error = 244.64

Figure 16: Error histograms of our four models considering the Tesla stock prices from 08/26/2019 to 02/24/2020

Figure (16) is very similar to what we have seen for the other plots meaning that our fourth model is the best and the third model performs the worst.

To summarize, we observe that overall our fourth model performs the best, followed by our first model, then our second model and finally our third model. We can draw several conclusions by this order. The most important is that the mean SFDE from model 3 is not a good model for the stock price, especially for long term projections. This leads to the next important aspect. When modelling stock prices it is necessary to have a parameter or something else to model stock prices over a long period of time, for example the simplest thing we can do is introducing the term $\mu X(t)$ (as in model 1, 2, 4) which models a percentage change compared to the current value. However, we see that for short term modelling model 3 holds up relatively well and additionally we observe that our fourth model is the best in

almost all cases which suggests that there indeed exists some "evolving around its mean" behaviour of stocks.

Finally, we observe that our first model performs significantly better than our second model although the models are based on the same SDE and we just estimated the parameters differently. It makes sense that the parameters estimated based on the Euler scheme perform better than the theoretical ones, since we are simulating paths with the Euler scheme afterwards and the parameters from model 1 are optizmised for exactly that.

5 Option Pricing for path-dependent volatility models

In this chapter we are going to use the SFDE from chapter 2 as a volatility model similarly to the Heston model. We are going to look at a numerical scheme to find option prices of any kind for our model and compare those results to other models.

Previously, in chapter 4 we used an SFDE which is evolving around its mean for modelling the stock market. We now present a model where the volatility is modelled by its own SFDE and is not assumed constant anymore. The model reads as

$$\begin{aligned} dX(t) &= \mu X(t)dt + \sigma(t)X(t)dW^X(t), \text{ where } \sigma(t) = f(Z(t)) \text{ and} && (37)\\ dZ(t) &= \kappa(\bar{Z}_t - Z(t))dt + \xi dW^Z(t), && (38) \end{aligned}$$

where $\mu \in \mathbb{R}, \kappa > 0, \xi > 0$, $W^X(t)$ and $W^Z(t)$ are Wiener processes with correlation ρ and f is a differentiable function. We see that that the volatility is not constant anymore, but $\sigma(t)$ evolves now.

We observe that this model is similar to the Heston model which is given by

$$\begin{aligned} dX(t) &= \mu X(t)dt + \sigma(t)X(t)dW^X(t), \text{ where } \sigma(t) = f(Z(t)) \text{ and} && (39)\\ dZ(t) &= \kappa(\theta - Z(t))dt + \xi dW^Z(t), && (40) \end{aligned}$$

with the same assumptions as the previous model and additionally $\theta \in \mathbb{R}$.

Next, we are going to present the Euler scheme for the model from equations (38) and (40). Therefore, we need to simulate two normal random variables with a given correlation coefficient. We know that $\Delta W \sim \mathcal{N}(0, \Delta)$ holds. Hence, we first simulate two independent random variables $X \sim \mathcal{N}(0, \Delta)$ and $Y \sim \mathcal{N}(0, \Delta)$. Afterwards, we set $Z := \rho X + \sqrt{1-\rho^2}Y$. Then it holds $Z \sim \mathcal{N}(0, \Delta)$ as well as

$$\begin{aligned} \text{Cov}(X, Z) &= \text{Cov}(X, \rho X + \sqrt{1-\rho^2}Y) = \text{Cov}(X, \rho X) + \text{Cov}(X, \sqrt{1-\rho^2}Y)\\ &= \rho\text{Cov}(X, X) = \rho\text{Var}(X) = \rho\Delta, \end{aligned}$$

since X and Y are independent. Therefore,

$$\mathrm{Cor}(X,Z) = \frac{\mathrm{Cov}(X,Z)}{\sqrt{\mathrm{Var}(X)\cdot\mathrm{Var}(Z)}} = \frac{\rho\Delta}{\sqrt{\Delta\cdot\Delta}} = \rho.$$

We see that now X and Z are going to be our realizations of our correlated Wiener processes, since both of them are $\mathcal{N}(0,\Delta)$ distributed and have the desired correlation.

We are now able to write down the Euler scheme for our two models from this chapter. We note that the models from equations (38) and (40) are incomplete models, hence the Euler scheme is not providing us the unique fair value of an option, but rather a value in the non-arbitrage range. We start with the Euler scheme for the model from equation (38). It reads as

$$\begin{aligned} Y_{n+1} &= Y_n + \mu Y_n\Delta + f(\nu_n)Y_n\Delta W^Y, Y_0 = X_0 \\ \nu_{n+1} &= \nu_n + \kappa(\bar{\nu}_n - \nu_n)\Delta + \xi\Delta W^\nu, \nu_0 = Z_0. \end{aligned}$$

The Euler scheme for the model from equation (40) looks very similar. It is given by

$$\begin{aligned} Y_{n+1} &= Y_n + \mu Y_n\Delta + f(\nu_n)Y_n\Delta W^Y, Y_0 = X_0 \\ \nu_{n+1} &= \nu_n + \kappa(\theta - \nu_n)\Delta + \xi\Delta W^\nu, \nu_0 = Z_0. \end{aligned}$$

In both schemes ΔW^Y and ΔW^ν are realized as stated above. We want to take a look at option pricing for these two models. Therefore, we fix the parameters in the models and simulate the underyling paths by using the Euler schemes. As option we choose a basic call option where the fair price of an option is given by

$$\text{Price of a call option} = \exp(-rT)\mathbb{E}(\max\{0, S_T - K\}|S_0 = y), \tag{41}$$

where r is the risk free interest rate, T is the maturity of the option in years, S is the price of the underlying asset and y is the current value of the stock.

For our simulation we set $f(x) = \exp(x), S_0 = 100, K = 100, r = \mu = 0.1, \rho = 0.3, \kappa = 1, T = 2, \xi = 0.1, N = 100, \theta = -1.5$ and $\nu_0 = -1.5$. We have to choose $\theta = \nu_0$ to compare the results correctly, since we saw in chapter 2 that the the volatility modeling SFDE from equation (38) evolves around ν_0 and the volatility modeling SDE from equation (40) evolves around θ. So in order to get comparable results for the option price and the volatility the two volatility models have to evolve around the same value.

As a reference for both models we use the Black-Scholes model with the Euler scheme with fitting parameters and the Black-Scholes pricing formula. In order to obtain the price of the call option for each model we simulate 100000 paths of the underlying SFDE and check for each path seperately what the fair price would have been by calculating $\exp(-rT)\max\{0, S_T - K\}$. Afterwards, we use Monte Carlo simulation to approximate the expectation given in equation (41). The corresponding R-code can be found in the appendix (program 6). Using the fixed parameters from above, yields:

For model from equation (38): 22.34

For model from equation (40): 22.31

For Black-Scholes model with Euler scheme: 22.55

For Black-Scholes pricing formula: 22.65

We see that the prices do not differ very much from each other which is reasonable, since a model should not be really far away from the Black-Scholes model and all four methods are closely related. We are not going to present more examples with different parameters, but our simulations show that the relative distance between the four options differs not much, even for other option types.

A better approach for testing how good our two models introduced in this chapter really are, would be to calibrate the two models with real data from the stock market and afterwards do testing. Sadly, this is not an easy task and we neither had the time nor the data to perform such a calibration.

6 References

[KP95] Peter E Kloeden and Eckhard Platen. *Numerical solution of stochastic differential equations*, volume 2. Springer Science & Business Media, 1995.

[KT81] Samuel Karlin and Howard E Taylor. *A second course in stochastic processes*. Elsevier, 1981.

[Mao03] X. Mao. *Numerical solutions of stochastic functional differential equations*. London Mathematical Society, 2003.

[Øks03] Bernt Øksendal. Stochastic differential equations. In *Stochastic differential equations*. Springer, 2003.

Appendix R-Code

Program 1

```
n <- 1000
theta1 <- 1
sigma <- 0.2
start <- 500

y600 <- rep(0, 10000)
y800 <- rep(0, 10000)
y999 <- rep(0, 10000)
y1000 <- rep(0, 10000)
process <- c()
process[1] <- start
T <- 100
h <- T/n

for (i in 1:10000) {
  process <- c()
  process[1] <- start

  for (k in 1:(n-1)) {
    process[k+1] <- process[k]
        + theta1 * (mean(process)-process[k])*h
        + sigma * rnorm(1, 0, sqrt(h))

    }
```

```
25
26    y600[i] <- process[600]
27    y800[i] <- process[800]
28    y999[i] <- process[999]
29    y1000[i] <- process[1000]
30
31  }
32
33  hist(y600, main = "Histogram of values at time step 600",
34  ylab = "Frequency", xlab = "value",  xlim = c(499.7, 500.3),
35  ylim = c(0, 2500))
36  hist(y800, main = "Histogram of values at time step 800",
37  ylab = "Frequency", xlab = "value",  xlim = c(499.7, 500.3),
38  ylim = c(0, 2500))
39  hist(y999, main = "Histogram of values at time step 999",
40  ylab = "Frequency", xlab = "value",  xlim = c(499.7, 500.3),
41  ylim = c(0, 2500))
42  hist(y1000, main = "Histogram of values at time step 1000",
43  ylab = "Frequency", xlab = "value",  xlim = c(499.7, 500.3),
44  ylim = c(0, 2500))
```

Program 2

```
1  n <- 1000
2  theta1 <- 1
3  sigma <- 0.2
4  start <- 500
```

```
5
6  y600 <- rep(0, 10000)
7  y800 <- rep(0, 10000)
8  y999 <- rep(0, 10000)
9  y1000 <- rep(0, 10000)
10 process <- c()
11 process[1] <- start
12 T <- 100
13 h <- T/n
14
15 for (i in 1:10000) {
16   process <- c()
17   process[1] <- start
18
19   for (k in 1:(n-1)) {
20     process[k+1] <- process[k]
21         + theta1 * (mean(process)-process[k])*h
22         + sigma * process[k] * rnorm(1, 0, sqrt(h))
23
24   }
25   y600[i] <- process[600]
26   y800[i] <- process[800]
27   y999[i] <- process[999]
28   y1000[i] <- process[1000]
29
30 }
31
```

```
hist(y600, main = "Histogram of values at time step 600",
ylab = "Frequency", xlab = "value",  xlim = c(300, 700),
ylim = c(0, 2500))
hist(y800, main = "Histogram of values at time step 800",
ylab = "Frequency", xlab = "value",  xlim = c(300, 700),
ylim = c(0, 2500))
hist(y999, main = "Histogram of values at time step 999",
ylab = "Frequency", xlab = "value",  xlim = c(300, 700),
ylim = c(0, 2500))
hist(y1000, main = "Histogram of values at time step 1000",
ylab = "Frequency", xlab = "value",  xlim = c(300, 700),
ylim = c(0, 2500))
```

Program 3: Here for the OU process, the R-code for the OU related proces is similar

```
n <- 1000
theta1 <- 1
sigma <- 0.4
start <- 500
y5 <- rep(0, 10001)
y6 <- rep(0, 10001)
y50 <- rep(0, 10001)
y999 <- rep(0, 10001)
y1000 <- rep(0, 10001)
mu <- 1
process <- c()
process[1] <- start
```

```
T <- 100
h <- T/n

for (i in 1:10001) {
  process <- c()
  process[1] <- start

  for (k in 1:(n-1)) {
    process[k+1] <- process[k]
    + theta1 * (mu-process[k])*h
    + sigma * rnorm(1, 0, sqrt(h))

  }
  y5[i] <- process[5]
  y6[i] <- process[6]
  y50[i] <- process[50]
  y999[i] <- process[999]
  y1000[i] <- process[1000]

}


a <- (2*theta1*mu)/(sigma*sigma)
b <- theta1/(sigma^2)
x <- seq(0, 2, 0.0002)
C <- 2295.384564755
truedens <- (1/(sigma^2))*(1/C)*exp(a*x-b*x^2)
```

```
40
41
42 plot(x, truedens, col= "red", xlim= c(0.2, 1.8)
43 , main= "Densities ", ylab = "f(x)")
44 lines(density(y999), col = "blue")
45 lines(density(y1000))
```

Program 4: Specific Code for simulating the paths for the OU related process, the R-codes for the two SFDE's and the OU process are similar

```
1  n <- 1001
2  theta1 <- 1
3  sigma <- 0.2
4  start <- 500
5  m <- 1001
6  mu <- 300
7  process <- c()
8  process[1] <- start
9  T <- 100
10 h <- T/n
11 x <- seq(0, 100, 0.1)
12 P <- matrix(0, m, n)
13 P[,1]=start
14
15
16 for (i in 1:m) {
17   process <- rep(0, 1001)
```

```
  process[1] <- start

  for (k in 1:(n-1)) {
    process[k+1] <- process[k]
    + theta1 * (mu-process[k])*h
    + sigma * process[k] * rnorm(1, 0, sqrt(h))
    P[i, k+1] = process[k+1]
  }

}



middleprocess <- rep(0, n)
for (j in 1:(n)) {
  middleprocess[j] <- mean(P[2:m,j])
}


plot(x, middleprocess, type = "l", col = "blue", xlab = "t"
, ylab = "Process", ylim = c(200, 550))
lines(x, P[1,], col = "red")
```

Program 5: One example of R-Code evaluating our models for stocks (in this case we use the one month Amazon data). The other codes used in chapter 4 are the same except for the given data.

```
rm(list = ls())
setwd("C:/Users/Laszlo/Desktop/MA/Datenauswertung/Data")
data = read.csv("Amazon1M.csv")
x <- data[,2]
n <- length(x)
#assumption 252 trading days, we have 21 days
T <- n/252
h <- T/n
#eulerBS, parameters: myu1, sigma1
myuvec1 <- rep(0, n-1)
sigmavec1 <- rep(0, n-1)
for (k in 1:(n-1)) {
  myuvec1[k] <- (x[k+1]-x[k])/(x[k]*h)
}
myu1 <- mean(myuvec1)
for (k in 1:(n-1)) {
  sigmavec1[k] <-(abs(x[k+1]-x[k]-myu1*x[k]*h))
                 /(x[k]*sqrt((2*h)/pi))
}
sigma1 <- mean(sigmavec1)

#eulertheoretical, Parameters: myu2, sigma2
logreturn <- rep(0, n-1)
for (i in 1:(n-1)) {
  logreturn[i] <- log(x[i+1]/x[i])
}
mean <- mean(logreturn)
```

```
std <- sd(logreturn)
meanannualized <- 252*mean
stdannualized <- std*sqrt(252)
myu2 <- meanannualized + (stdannualized^2)/2
sigma2 <- stdannualized

#MeanSDE, parameters sigma
meanprocess <- rep(0,n)
meanprocess[1] <- x[1]

for (i in 2:n) {
  meanprocess[i] <- (meanprocess[i-1]*(i-1)+x[i])/i
}
sigmavec3 <- rep(0,n-1)
for (k in 1:(n-1)) {
  sigmavec3[k] <- abs((x[k+1]-x[k]
        -(meanprocess[k]-x[k])*h)/x[k])
}
sigma3 <- mean(sigmavec3)/sqrt((2*h)/pi)

#MeanSDE2, parameters sigma, myu
myuvec4 <- rep(0, n-1)
sigmavec4 <- rep(0, n-1)
for (k in 1:(n-1)) {
  a <- (x[k+1]-x[k])/h
  b <- a - meanprocess[k]+x[k]
  myuvec4[k] <- b/x[k]
```

```
}
myu4 <- mean(myuvec4)

for (k in 1:(n-1)) {
  sigmavec4[k] <-
        (abs(x[k+1]-x[k]-(meanprocess[k]-x[k]+myu4*x[k])*h))
        /(x[k]*sqrt((2*h)/pi))
}
sigma4 <- mean(sigmavec4)


#eulerBS error estimation
error1 <- rep(0, 10000)

for (k in 1:10000) {
  process1 <- rep(0, n)
  process1[1] <- x[1]
  for (i in 2:n) {
    process1[i] <- process1[i-1]
            + myu1*process1[i-1] * h
        + sigma1*process1[i-1]*rnorm(1, 0, sqrt(h))
  }
  error1[k] <- max(abs(x-process1))
}

#eulertheoretical error estimation
error2 <- rep(0, 10000)
```

```
82
83  for (k in 1:10000) {
84    process2 <- rep(0, n)
85    process2[1] <- x[1]
86    for (i in 2:n) {
87      process2[i] <- process2[i-1]
88                + myu2*process2[i-1] * h
89           + sigma2*process2[i-1]*rnorm(1, 0, sqrt(h))
90    }
91    error2[k] <- max(abs(x-process2))
92
93  }
94  #eulermeanSDE error estimation
95  error3 <- rep(0, 10000)
96
97  for (k in 1:10000) {
98    process3 <- c()
99    process3[1] <- x[1]
100   for (i in 2:n) {
101     process3[i] <- process3[i-1]
102               +(mean(process3)-process3[i-1])*h
103               +sigma3*process3[i-1]*rnorm(1, 0, sqrt(h))
104   }
105   error3[k] <- max(abs(x-process3))
106 }
107
108 #eulermeanSDE2 error estimation
```

```
109 error4 <- rep(0, 10000)
110
111 for (k in 1:10000) {
112   process4 <- c()
113   process4[1] <- x[1]
114   for (i in 2:n) {
115     process4[i] <- process4[i-1]
116         +(mean(process4)-process4[i-1]+myu4*process4[i-1])*h
117         +sigma4*process4[i-1]*rnorm(1, 0, sqrt(h))
118   }
119   error4[k] <- max(abs(x-process4))
120 }
121
122 hist(error1, main = "Error for Black-Scholes with Euler",
123 xlab = "Error from the true path", ylab = "Frequency",
124 xlim = c(0, 400), ylim = c(0, 2800))
125 hist(error2, main = "Error for Black-Scholes theoretical",
126 xlab = "Error from the true path", ylab = "Frequency",
127 xlim = c(0, 400), ylim = c(0, 2800))
128 hist(error3, main = "Error for MeanSDE with Euler",
129 xlab = "Error from the true path", ylab = "Frequency",
130 xlim = c(0, 400), ylim = c(0, 2800))
131 hist(error4, main = "Error for MeanSDE2 with Euler",
132 xlab = "Error from the true path", ylab = "Frequency",
133 xlim = c(0, 400), ylim = c(0, 2800))
134
135 meanerror1 <- mean(error1)
```

```
meanerror2 <- mean(error2)
meanerror3 <- mean(error3)
meanerror4 <- mean(error4)
```

Program 6

```
n <- 100
S <- rep(0, n)
S[1] <- 100
nyu <- c()
nyu[1] <- -1.5
r <- 0.1
rho <- 0.3
T <- 2
kappa <- 1
xi <- 0.1
h <- T/n
strike <- 100

estimate1 <- rep(0, 100000)
for (k in 1:100000) {
  nyu <- c()
  nyu[1] <- -1.5
  S <- rep(0, n)
  S[1] <- 100
   x1<-rnorm(100,0,sqrt(h))
  y1<-x1*rho+sqrt(1-rho^2)*rnorm(100,0,sqrt(h))
```

```
for (i in 2:n) {
  S[i] <- S[i-1]+r*S[i-1]*h+exp(nyu[i-1])*S[i-1]*x1[i]
  nyu[i] <- nyu[i-1]+kappa*(-nyu[i-1]+mean(nyu))*h+xi*y1[i]
}
estimate1[k] <- max(0, S[n]-strike)
}
callprice1 <- mean(estimate1)*exp(-r*T)

estimate2 <- rep(0, 100000)
for (k in 1:100000) {
  nyu <- c()
  nyu[1] <- -1.5
  S <- rep(0, n)
  S[1] <- 100
  x2<-rnorm(100,0,sqrt(h))
  y2<-x2*rho+sqrt(1-rho^2)*rnorm(100,0,sqrt(h))
  for (i in 2:n) {
    S[i] <- S[i-1]+r*S[i-1]*h+exp(nyu[i-1])*S[i-1]*x2[i]
    nyu[i] <- nyu[i-1]+kappa*(-nyu[i-1]+nyu[1])*h+xi*y2[i]
  }
  estimate2[k] <- max(0, S[n]-strike)
}
callprice2 <- mean(estimate2)*exp(-r*T)

estimate3 <- rep(0, 100000)
for (k in 1:100000) {
  S <- rep(0, n)
```

```
49    S[1] <- 100
50    for (i in 2:n) {
51      S[i] <- S[i-1]+r*S[i-1]*h+
52           exp(nyu[1])*S[i-1]*rnorm(1, 0, sqrt(h))
53
54    }
55    estimate3[k] <- max(0, S[n]-strike)
56  }
57  callprice3 <- mean(estimate3)*exp(-r*T)
```