Continous Time Finance 2 - Hand-in 2

Clara E. Tørsløv (cnp777)

23. marts 2021

Throughout this assignment references to 'Björk' is a reference to Tomas Björk, *Arbitrage Theory in Continuous Time*, third edition (2009).

1 The Bachelier Model

The Bachelier model models the price of a stock, S, by an arithmetic process, which has dynamics of the form

$$dS(T) = ...dt + \sigma dW(t)$$

where σ is a constant, i.e. it does not have $S(t)$ itself in it.

1.a

We assume the interest rate is zero and consider a strike-K expiry-T call-option. First, we wish to show that its arbitrage-free time-t price is

$$\pi^{call,Bach}(t) = (S(t) - K)\Phi\left(\frac{S(t) - K}{\sigma\sqrt{T-t}}\right) + \sigma\sqrt{T-t}\phi\left(\frac{S(t) - K}{\sigma\sqrt{T-t}}\right)$$

Where Φ and ϕ denote, respectively, the standard normal distribution and density function. We start by noting that, as $r = 0$, the $\mathbb{Q}$-martingale dynamics becomes

$$dS(t) = \sigma dW^{\mathbb{Q}}(t)$$

This integrates to be

$$\begin{aligned} S(T) &= S(t) + \int_t^T \sigma dW(s) \\ &= S(t) + \sigma \cdot (W(T) - W(t)) \sim \mathcal{N}(S(t), \sigma^2(T-t)) \end{aligned}$$

Thus $S(T) - K \sim \mathcal{N}(S(t) - K, \sigma^2(T-t))$. By Risk-Neutral Valuation (with $r = 0$) we find

$$\begin{aligned}\pi^{call,Bach}(t) &= \mathbb{E}_t^{\mathbb{Q}}\left[(S(T)-K)^+\right] \\ &= \mathbb{E}_t^{\mathbb{Q}}\left[(S(T)-K)\mathbb{1}_{0\leq S(T)-K\leq\infty}\right]\end{aligned}$$

We now recall that for $X \sim \mathcal{N}(\mu, \sigma^2)$ it holds $\mathbb{E}[X\mathbb{1}_{l\leq X\leq h}] = \mu\left(\Phi\left(\frac{h-\mu}{\sigma}\right) - \Phi\left(\frac{l-\mu}{\sigma}\right)\right) + \sigma\left(\phi\left(\frac{l-\mu}{\sigma}\right) - \phi\left(\frac{h-\mu}{\sigma}\right)\right)$. Using that $S(T) - K \sim \mathcal{N}(S(t) - K, \sigma^2(T-t))$ and slightly abusing some 'infinity'-notation for ease

$$= (S(t)-K)\left(\Phi(\infty) - \Phi\left(\frac{-(S(t)-K)}{\sigma\sqrt{T-t}}\right)\right) + \sigma\sqrt{T-t}\left(\phi\left(\frac{-(S(t)-K)}{\sigma\sqrt{T-t}}\right) - \phi(\infty)\right)$$

We now use the following facts surrounding density functions: $\Phi(\infty) = 1$, $\phi(\infty) = 0$, $\phi(x) = \phi(-x)$ and $\Phi(-x) = 1 - \Phi(x)$.

$$\begin{aligned}&= (S(t)-K)\left(1 - \left(1 - \Phi\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right)\right)\right) + \sigma\sqrt{T-t}\cdot\phi\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right) \\ &= (S(t)-K)\Phi\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right) + \sigma\sqrt{T-t}\cdot\phi\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right)\end{aligned}$$

Which was what we wanted. Second, we wish to derive the time-t Δ-hedge-ratio of the call-option.

$$\Delta^{call,Bach}(t) = \frac{\partial\pi^{call,Bach}(t)}{\partial S(t)}$$

Using the chain-rule and the product rule we find

$$= \Phi\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right) + (S(t)-K)\Phi'\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right)\frac{1}{\sigma\sqrt{T-t}} + \sigma\sqrt{T-t}\phi'\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right)\frac{1}{\sigma\sqrt{T-t}}$$

We now use the fact that $\Phi' = \phi$ and $\phi'(x) = -x\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}x^2} = -x\phi(x)$

$$\begin{aligned}&= \Phi\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right) + \frac{(S(t)-K)}{\sigma\sqrt{T-t}}\phi\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right) - \frac{(S(t)-K)}{\sigma\sqrt{T-t}}\phi\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right) \\ &= \Phi\left(\frac{S(t)-K}{\sigma\sqrt{T-t}}\right)\end{aligned}$$

1.b

In this problem we wish to examine what the implied volatilities across strikes look like in the Bachelier model.[1] The code can be found in the Appendix and the result is shown in figure 1. The implied volatility

[1] Implied volatility is defined in section 7.7.2 in Björk

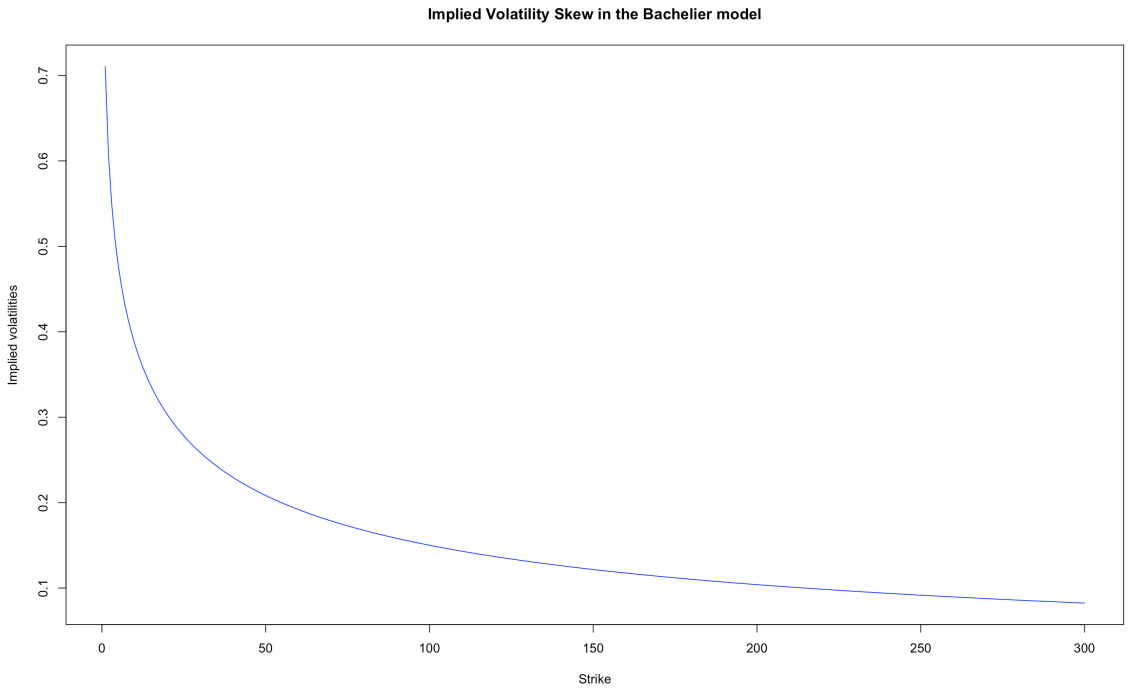
at time t is determined by solving for σ^{IV} in the following equation

$$\pi^{call,Bach}(t) = Call^{BS}(S(t), t, T, r, \sigma^{IV}, K)$$

Here $\pi^{call,Bach}(t)$ denotes the market price, i.e. the benchmark option, which in our case is determined by the pricing formula from 1.a. $\pi^{call,Bach}(t)$ is the option price under the assumption that the Bachelier model is the correct way to model stock prices. Running this price backwards through the Black-Scholes equation is then a way of evaluating this price under the assumption that the Black Scholes model is the correct way to model stock prices. Intuitively this entails that a high implied volatility would mean that the market price, $\pi^{call,Bach}(t)$, is considered high under the Black Scholes model and vice versa. Looking at figure 1 we clearly see a so-called reverse skew. This means that the market price, i.e. the Bachelier price, of ITM calls is considered high relative to the Black-Scholes and reversely.

Noteworthy differences in the two models can be found in the diffusion terms. These differ in the way that the volatility in the Bachelier model is the volatility of the return itself, whereas the volatility in the Black-Scholes model is the volatility of the rate of return. Further the distribution of the underlying asset differ. The stock is assumed to follow a log-normal distribution in the Black-Scholes model and a normal distribution in the Bachelier model. This often entails that the Black-Scholes implied volatility is referred to as the implied log-normal volatility and the Bachelier implied volatility as the implied normal volatility. This is further examined in the SABR model (using $\beta = 1$ and $\beta = 0$ for log-normal and normal implied volatility respectively)[2], where Hagan et. al.[3] finds that $\beta = 0$ gives a steeply downward sloping backbone, i.e. ATM implied volatility curve, and the exponent $\beta = 1$ gives an almost flat backbone. This is exactly what can be observed in figure 1.

[2]Conditioned on a realization of the volatility, which is assumed constant in the Bachelier and Black-Scholes model.
[3]Hagan P., et. al. (2002), "Managing smile risk", Wilmott Journal, page 90.



Figur 1: Plot of implied volatilities in the Bachelier model using parameters: $S_0 = 100$, $T = 1$ and $\sigma = 15$.

1.c

In this problem we wish to examine what the call-price formula looks like in a Bachelier model with a non-zero (but constant) interest rate r and $\mathbb{Q}$-dynamics given by

$$dS(t) = rS(t)dt + \sigma dW^{\mathbb{Q}}(t)$$

We introduce the discounted stock, $\tilde{S}(t) = e^{-rt}S(t)$, which has the following dynamics (by Ito)

$$d\tilde{S}(t) = -re^{-rt}S(t)dt + e^{-rt}dS(t) = -r\tilde{S}(t)dt + e^{-rt}(rS(t)dt + \sigma dW^{\mathbb{Q}}(t)) = e^{-rt}\sigma dW^{\mathbb{Q}}(t)$$

From this it follows, by Björk Proposition 5.3, that

$$\tilde{S}(T) = \tilde{S}(t) + \int_t^T e^{-rs}\sigma dW^{\mathbb{Q}}(s) \Rightarrow S(T) = e^{r(T-t)}S(t) + \sigma \int_t^T e^{r(T-s)}dW^{\mathbb{Q}}(s)$$

By Björk Lemma 4.15 we know the distribution of $S(T)$ and further the distribution of

$$\begin{aligned} S(T) - K &\sim \mathcal{N}\left(e^{r(T-t)}S(t) - K, \sigma^2 \int_t^T e^{2r(T-s)}ds\right) \\ &= \mathcal{N}\left(e^{r(T-t)}S(t) - K, \sigma^2[-\frac{1}{2r}e^{2r(T-s)}]_t^T\right) \\ &= \mathcal{N}\left(e^{r(T-t)}S(t) - K, \underbrace{\frac{\sigma^2}{2r}(e^{2r(T-t)} - 1)}_{:=\tilde{\sigma}^2}\right) \end{aligned}$$

where the variance, $\tilde{\sigma}^2$, has been defined for ease of notation. Following exactly the same steps as in 1.a, the call price is found

$$\begin{aligned} \pi^{call,Bach}(t) &= e^{-r(T-t)}\mathbb{E}_t^{\mathbb{Q}}\left[(S(T) - K)^+\right] \\ &= e^{-r(T-t)}\mathbb{E}_t^{\mathbb{Q}}\left[(S(T) - K)\mathbb{1}_{0\leq S(T)-K\leq\infty}\right] \\ &= e^{-r(T-t)}(e^{r(T-t)}S(t) - K)\left(\Phi(\infty) - \Phi\left(\frac{-(e^{r(T-t)}S(t) - K)}{\tilde{\sigma}}\right)\right) \\ &+ e^{-r(T-t)}\tilde{\sigma}\left(\phi\left(\frac{-(e^{r(T-t)}S(t) - K)}{\tilde{\sigma}}\right) - \phi(\infty)\right) \\ &= e^{-r(T-t)}(e^{r(T-t)}S(t) - K)\Phi\left(\frac{e^{r(T-t)}S(t) - K}{\tilde{\sigma}}\right) + e^{-r(T-t)}\tilde{\sigma}\phi\left(\frac{-(e^{r(T-t)}S(t) - K)}{\tilde{\sigma}}\right) \\ &= (S(t) - e^{-r(T-t)}K)\Phi\left(\frac{e^{r(T-t)}S(t) - K}{\frac{\sigma^2}{2r}(e^{2r(T-t)} - 1)}\right) \\ &+ \frac{\sigma}{\sqrt{2r}}\sqrt{1 - e^{-2r(T-t)}} \cdot \phi\left(\frac{-(e^{r(T-t)}S(t) - K)}{\frac{\sigma^2}{2r}(e^{2r(T-t)} - 1)}\right) \end{aligned}$$

1.d

In this problem we wish to apply Dupire's formula to call-prices generated by the Bachelier model.

We again assume $r = 0$, then $dS(t) = \sigma dW^{\mathbb{Q}}(t) = \frac{\sigma}{S(t)}S(t)dW^{\mathbb{Q}}(t)$. In the Dupire model we assume risk-neutral dynamics of the underlying of the form $\frac{dS(t)}{S(t)} = \sigma(t, S(t))dW^{\mathbb{Q}}(t)$ where $\sigma(t, S(t))$ denotes the local volatility. Comparing these two stock dynamics we clearly see that $\sigma(S(t), t) = \frac{\sigma}{S(t)}$. As expressed in Gatheral equation (1.6)[4], Dupires formula states a definition of the local volatility function.

$$\sigma^2(K, T) = \frac{2Call_T^{Bach}(K, T)}{K^2 Call_{KK}^{Bach}(K, T)}$$

[4] Jim Gatheral, *The Volatility Surface*, 2006.

We now wish to find the analytical and numerical solutions to this formula and compare them. We start by finding the analytical, i.e. we wish to find the derivative of $Call^{Bach}$ w.r.t. T and the second derivative w.r.t. K. In the following we use the product rule, chain rule, $\Phi'(x) = \phi(x)$ and that $\phi'(x) = -x\phi(x)$. First note that

$$Call^{Bach}(K,T) = (S(0)-K)\Phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right) + \sigma\sqrt{T}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)$$

$$\begin{aligned}\frac{\partial Call^{Bach}}{\partial K} &= (-1)\cdot\Phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right) + (S(0)-K)\Phi'\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)\cdot\frac{-1}{\sigma\sqrt{T}} + \sigma\sqrt{T}\phi'\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)\cdot\frac{-1}{\sigma\sqrt{T}} \\ &= -\Phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right) - \frac{(S(0)-K)}{\sigma\sqrt{T}}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right) + \frac{(S(0)-K)}{\sigma\sqrt{T}}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right) \\ &= -\Phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)\end{aligned}$$

$$\frac{\partial^2 Call^{Bach}}{\partial K^2} = \frac{\partial}{\partial K}\left(-\Phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)\right) = -\Phi'\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)\cdot\left(-\frac{1}{\sigma\sqrt{T}}\right) = \frac{1}{\sigma\sqrt{T}}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)$$

$$\begin{aligned}\frac{\partial Call^{Bach}}{\partial T} &= (S(0)-K)\Phi'\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)\left(\frac{S(0)-K}{\sigma}\frac{-1}{2T^{3/2}}\right) + \sigma\frac{1}{2\sqrt{T}}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right) \\ &\quad + \sigma\sqrt{T}\phi'\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)\left(\frac{S(0)-K}{\sigma}\frac{-1}{2T^{3/2}}\right) \\ &= -\frac{1}{2}\frac{(S(0)-K)^2}{2T^{3/2}}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right) + \frac{1}{2}\frac{\sigma}{\sqrt{T}}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right) + \frac{1}{2}\frac{(S(0)-K)^2}{2T^{3/2}}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right) \\ &= \frac{\sigma}{2\sqrt{T}}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)\end{aligned}$$

Inserting these in to the Dupire formula we find

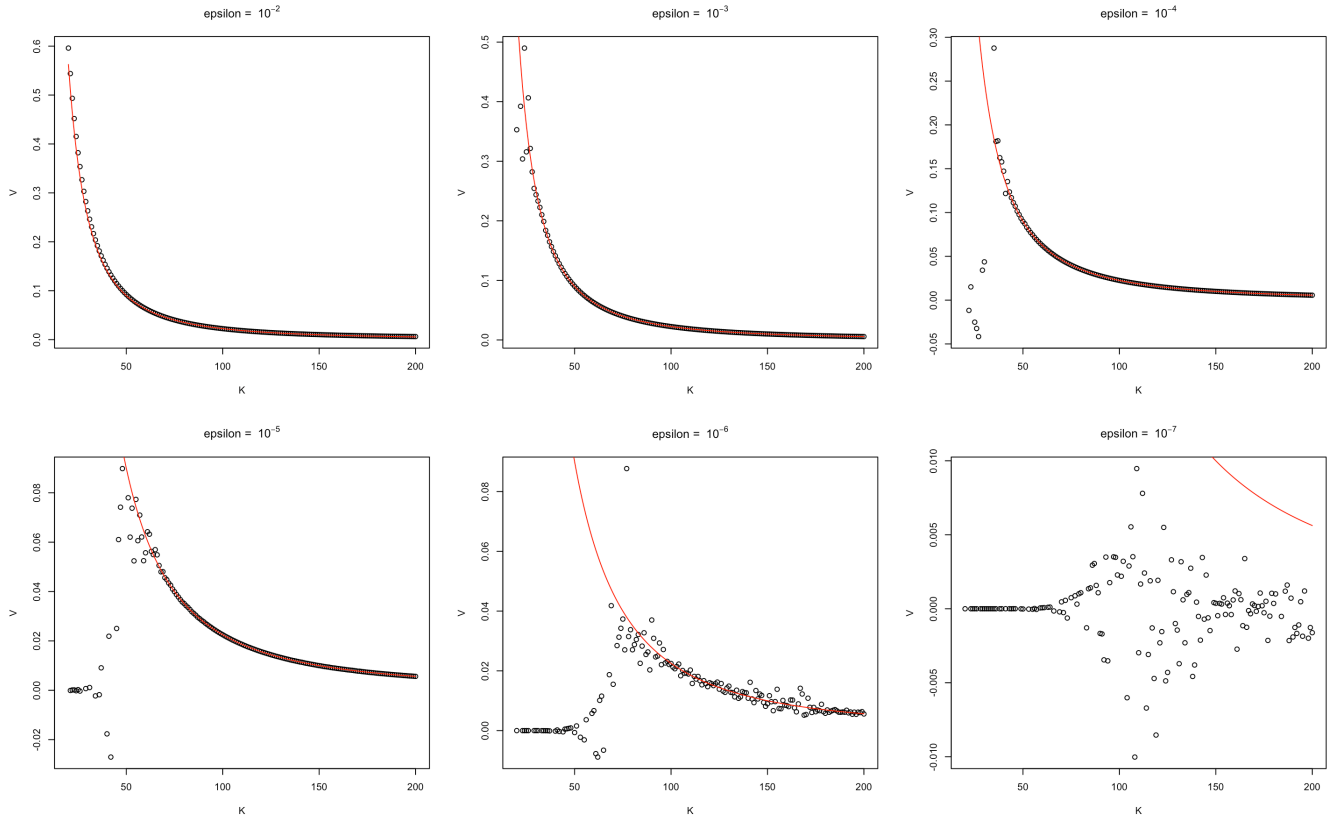
$$\sigma^2(K,T) = \frac{2\frac{\sigma}{2\sqrt{T}}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)}{K^2\frac{1}{\sigma\sqrt{T}}\phi\left(\frac{S(0)-K}{\sigma\sqrt{T}}\right)} = \frac{\sigma^2}{K^2} \Leftrightarrow \sigma(K,T) = \frac{\sigma}{K}$$

Which of course is of no surprise to us. To solve numerically we use the following standard derivative approximations

$$\begin{aligned}\frac{f(x+\epsilon)-f(x)}{\epsilon} \to f'(x) \text{ for } \epsilon\to 0 \Rightarrow \text{ So use } f'(x) = \frac{f(x+\epsilon)-f(x)}{\epsilon} \\ \frac{f(x+\epsilon)-2f(x)+f(x-\epsilon)}{\epsilon^2} \to f''(x) \text{ for } \epsilon\to 0 \Rightarrow \text{ So use } f''(x) = \frac{f(x+\epsilon)-2f(x)+f(x-\epsilon)}{\epsilon^2}\end{aligned}$$

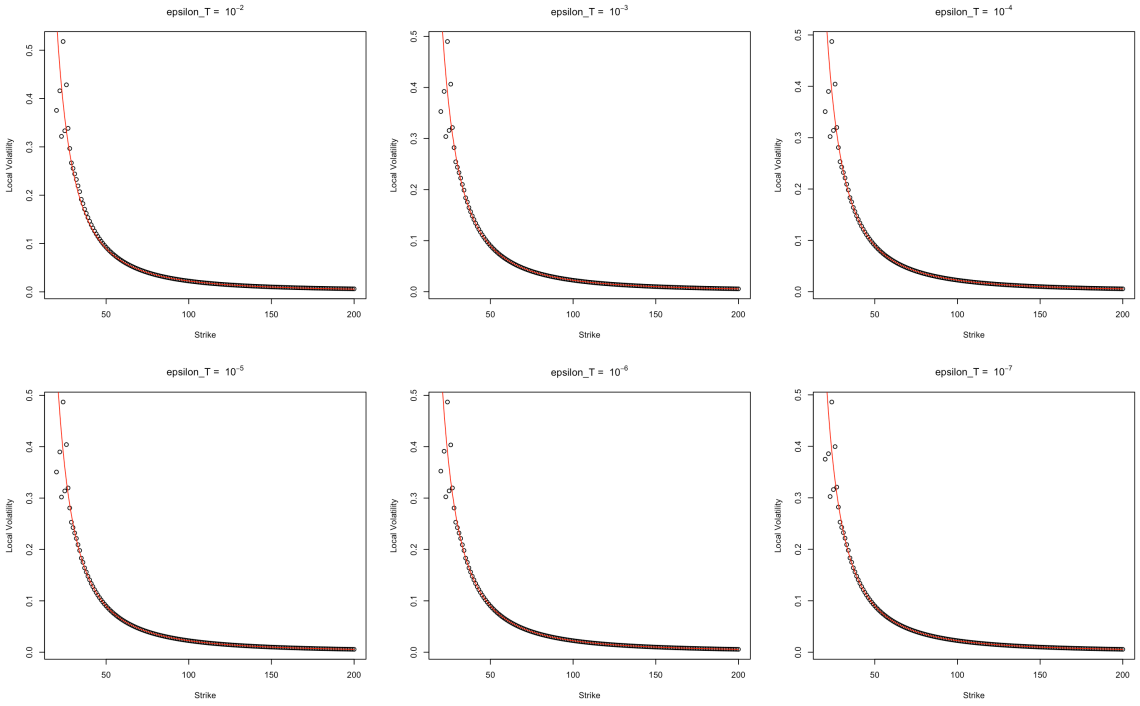
Using $f = \pi^{call,Bach}$, from 1.a, and plotting for different values of epsilon we find[5]

[5]For R-code see the Appendix.

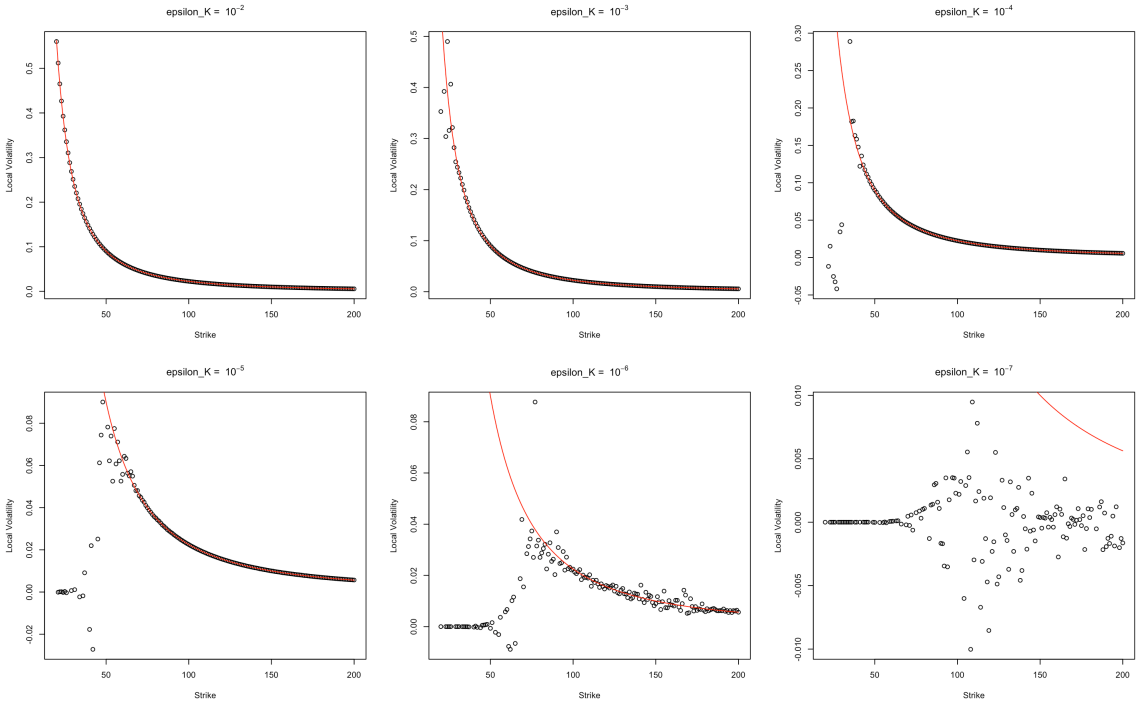


Figur 2: Local volatility squared by Dupires formula. The red line denotes the analytical solution and the black denotes numerical. Parameters: $S(0) = 100$, $\sigma = 15$, $T = 1$, $K = c(20 : 200)$.

We note from figure 2 that, as epsilon gets smaller, the analytical and numerical solutions' fit worsen increasingly, and for $\epsilon = 10^{-7}$ the fit is completely off. In addition, it seems that the worsening of the fit is more pronounced in the smaller strike values than in the larger. Thus, using the numerical estimation method we must be careful when choosing the value of epsilon. However, when using smaller and smaller epsilons we would have expected to move closer to the real value of the derivatives. We note how epsilon enters in both $\pi_T^{call,Bach}$ and $\pi_{KK}^{call,Bach}$. To further examine the effect we can let only one of the derivates at a time be epsilon-dependent and let epsilon be a constant in the other derivative. From figures 3 and 4 it is clearly seen that the negative effect on the fit of the squared local volatility for smaller and smaller epsilon stems from the double derivative w.r.t. K. Actually it looks as if there is no effect of decreasing epsilons on the $\pi_T^{call,Bach}$ at all.



Figur 3: Local volatility squared by Dupires formula with a constant epsilon in the $Call_{KK}$ function and different epsilons in the $Call_T$ derivative. Parameters: $S(0) = 100$, $\sigma = 15$, $T = 1$, $\epsilon_K = 10^{-3}$.



Figur 4: Local volatility squared by Dupires formula with a constant epsilon in the $Call_T$ function and different epsilons in the $Call_{KK}$ derivative. Parameters: $S(0) = 100$, $\sigma = 15$, $T = 1$, $\epsilon_T = 10^{-3}$.

2 Heston: Pricing and Hedging

In this problem we consider the stochastic volatility model, the Heston model, which has the following dynamics

$$dS(t) = rS(t)dt + \sqrt{\nu(t)}S(t)dW^{\mathbb{Q},1}(t) \tag{1}$$

$$d\nu(t) = \kappa(\theta - \nu(t))dt + \sigma\sqrt{\nu_t}dW^{\mathbb{Q},2}(t) \tag{2}$$

where $W^{\mathbb{Q},2}(t) = \rho W^{\mathbb{Q},1}(t) + \sqrt{1+\rho^2} \cdot \widetilde{W}(t)$ such that $dW^{\mathbb{Q},1}(t)dW^{\mathbb{Q},2}(t) = \rho dt$.[6] $\widetilde{W}(t)$, $W^{\mathbb{Q},2}(t)$ and $W^{\mathbb{Q},1}(t)$ are Brownian Motions where $\widetilde{W}(t) \perp\!\!\!\perp W^{\mathbb{Q},1}(t)$. In these dynamics $\sqrt{\nu(t)}$ replaces our usual Black-Scholes volatility and is interpretated as the instantaneous volatility of the stocks rate of return. Note that if $\sqrt{\nu(t)}$ was a constant we would be back in the Black-Scholes model. In the Heston model $\nu(t)$ is itself a stochastic process with its own Brownian Motion, meaning that we have an extra risk factor. The dynamics $d\nu(t)$ is recognized to be similar to the mean-reverting interest rate model, CIR.

2.a: Pricing

In this problem we wish to do simulation experiments in order to best price a european call option in the Heston model. To simulate we use the Euler discretization, meaning we update (S, ν) at each timestep in the following way

$$S(t+dt) = S(t) + rS(t)dt + \sqrt{\nu(t)}S(t)dW^{\mathbb{Q},1}$$
$$\nu(t+dt) = \nu(t) + \kappa(\theta - \nu(t))dt + \sqrt{\nu(t)}\sigma(\rho dW^{\mathbb{Q},1} + \sqrt{1-\rho^2}d\widetilde{W})$$

Note that this implies that negative values of $\nu(t)$ might be generated with non-zero probability, in which case computation of $\sqrt{\nu(t)}$ would be impossible. We therefore wish to analyze the following different discretization schemes to handle this problem. Each scheme will be explained further upon use.

- Absolute value
- Max-function
- Full truncation
- Moment matching
- Variance reduction by control variate

We present the results and analyze the effects in implied volatility space. We will do the following in R

[6] We write $W^{\mathbb{Q},2}(t)$ in this way for ease of simulation later.

for the five different schemes to produce 5 different plots:

1. We chose a discretization scheme.
2. We simulate (S, ν) paths.
3. We compute call prices using these paths.
4. We compute implied volatility using simulated option prices for different strikes.
5. We compute implied volatility using Heston closed-form solution call price for different strikes.
6. We plot the two implied volatilities in the same plot to compare and analyze.

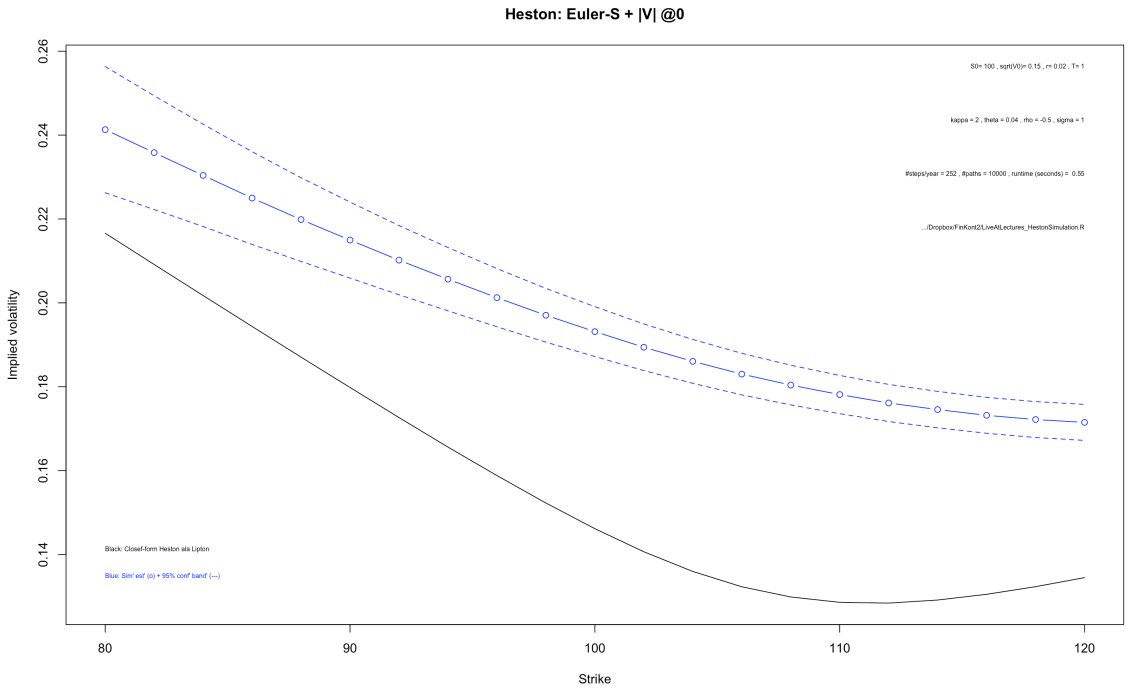
We use R-code from the lectures for the first three discretization schemes and the R-code for the latter two discretization schemes can be found in the Appendix.

Absolute value

In this case we update using the following scheme, handling negative values of $\nu(t)$ by taking the absolute value

$$\begin{aligned} S(t+dt) &= S(t) + rS(t)dt + \sqrt{\nu(t)}S(t)dW^{\mathbb{Q},1} \\ \nu(t+dt) &= \left|\nu(t) + \kappa(\theta - \nu(t))dt + \sqrt{\nu(t)}\sigma(\rho dW^{\mathbb{Q},1} + \sqrt{1-\rho^2}d\widetilde{W})\right| \end{aligned}$$

This scheme produces the following plot



Figur 5: Absolute value-discretization scheme.

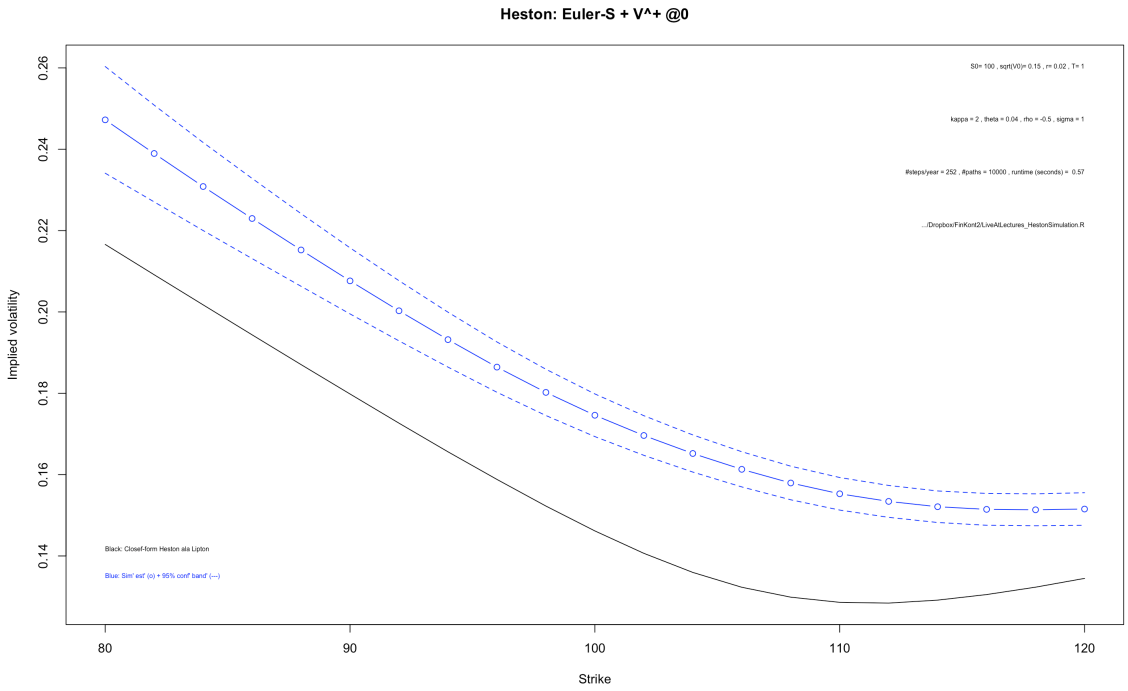
We note here that the implied volatility and confidence bounds derived from the simulated call price (blue curves) are well above the true curve (black). This error is more than just the usual Monte Carlo simulation discretization error. However we notice that the shape seems to be about right and the code runtime is very fast at 0.55 seconds.

Max function

In this case we update using the following scheme, handling negative values of $\nu(t)$ by letting it be equal to zero when it is negative

$$S(t+dt) = S(t) + rS(t)dt + \sqrt{\nu(t)}S(t)dW^{\mathbb{Q},1}$$
$$\nu(t+dt) = \max\left(\nu(t) + \kappa(\theta - \nu(t))dt + \sqrt{\nu(t)}\sigma(\rho dW^{\mathbb{Q},1} + \sqrt{1-\rho^2}d\widetilde{W}), 0\right)$$

This produces the following plot



Figur 6: Max-discretization scheme.

We note that the blue curves are closer but still well above the black curve. However the curvature and shape seems to match the true curve better. The runtime is 0.57 seconds.

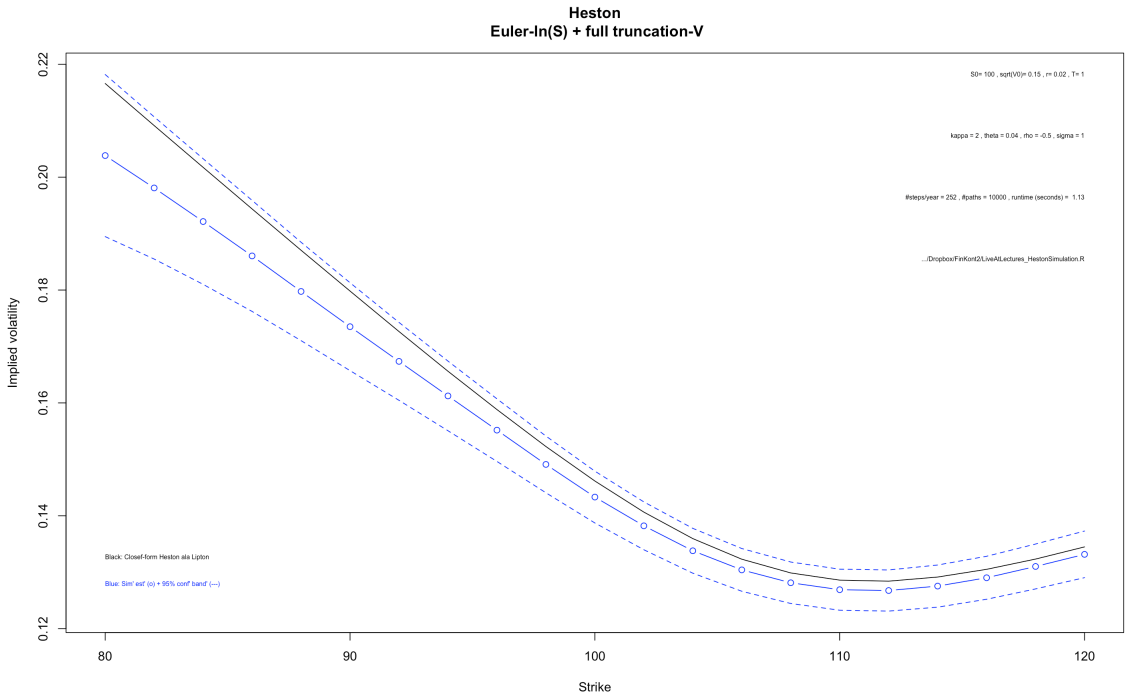
Full truncation

In this case we update using the following scheme, handling the issue of taking squareroot of negative values by replacing $\nu(t)$ with $\max(\nu(t), 0)$ on the right-hand-sides.[7] This scheme further uses a logarithmic discretization of the stock.

$$S(t+dt) = e^{\log(S(t))+(r-\frac{1}{2}\max(\nu(t),0))dt+\sqrt{\max(\nu(t),0)}dW^{\mathbb{Q},1}}$$
$$\nu(t+dt) = \nu(t) + \kappa(\theta - \max(\nu(t),0))dt + \sqrt{\max(\nu(t),0)}\sigma(\rho dW^{\mathbb{Q},1} + \sqrt{1-\rho^2}d\widetilde{W})$$

This produces the following plot

[7] From page 6 in https://www.dropbox.com/s/nw7uzmf8k0imq0t/LeifHestonWP.pdf?dl=0



Figur 7: Full truncation-discretization scheme.

We note how this is much better compared to the two previous plots, as the true curve is now completely inside the confidence bands. However, the simulated blue curve is still consistently below the black curve. Comparing the full-truncation scheme to the two previous (figures 5 and 6), we see that it does actually matter very much how we discretize.

The code runtime for above plot is 1.13 seconds. A snippet of the code used to simulate the paths for this plot is

```
dW1<-sqrt(dt)*rnorm(Nsim,0,1)
dW2<-rho*dW1+sqrt(1-rho^2)*sqrt(dt)*rnorm(Nsim,0,1)
X <- log(S)
X <- X + (R-0.5*pmax(V,0))*dt+sqrt(pmax(V,0))*dW1
S<-exp(X)
V<-V+kappa*(theta-pmax(V,0))*dt+sigma*sqrt(pmax(V,0))*dW2
```

We notice how these six lines of code could be decomposed into only three lines:

```
dW1<-sqrt(dt)*rnorm(Nsim,0,1)
S<-exp(log(S)+(R-0.5*pmax(V,0))*dt+sqrt(pmax(V,0))*dW1)
V<-V+kappa*(theta-pmax(V,0))*dt+
```

```
sigma*sqrt(pmax(V,0))*(rho*dW1+sqrt(1-rho^2)*sqrt(dt)*rnorm(Nsim,0,1))
```

Using the latter instead reduces the runtime to 0.8 seconds. This does not have a significant impact to us due to the small scale of our experiment but we might see a more notable difference if one were to have a higher simulation load.

Moment matching

The extention we wish to implement here is Leif Andersen's lognormal discretization as given in equation (29) in Lord, Koekkoek & van Dijk (2009). Here the problem of negative variance is fixed by making it locally lognormal distributed, i.e. positive with probability 1. Here we update in the following way

$$S(t+dt) = S(t)e^{+rS(t)dt+\sqrt{\max(\nu(t),0)}S(t)dW^{\mathbb{Q},1}(t)}$$
$$\nu(t+dt) = x \cdot e^{-\frac{1}{2}y^2dt+y\cdot dW^{\mathbb{Q},2}(t)}$$

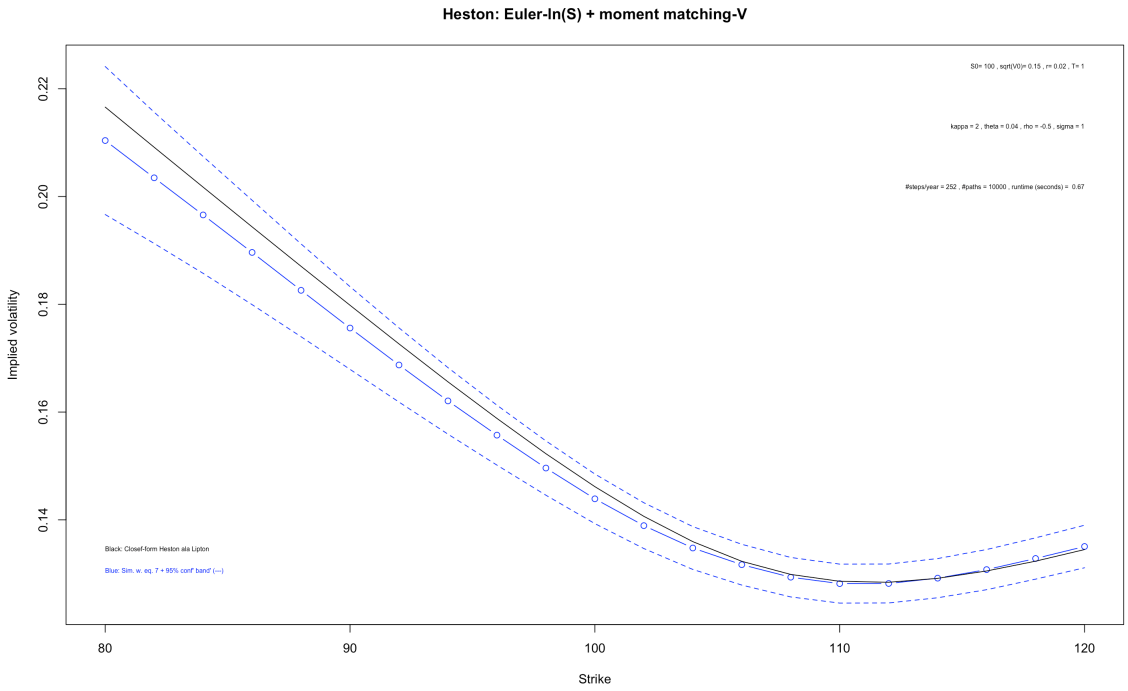
This discretization is build on moment-matching. We will not do the full-fledged derivation of the values x and y but merely scratch the surface for our own understanding.

The idea is that the conditional moments of $\nu(t+dt) = x \cdot e^{-\frac{1}{2}y^2+y\cdot Z}$, $Z \sim \mathcal{N}(0,1)$, are found: $\mathbb{E}_t[\nu^{LogNormal}(t+dt)]$ and $Var_t[\nu^{LogNormal}(t+dt)]$. Then the conditional moments of the CIR process in the Heston model are found: $\mathbb{E}_t[\nu^{Heston}(t+dt)]$ and $Var_t[\nu^{Heston}(t+dt)]$.[8] Then x and y are chosen such that these two moments match. Thus the discretization scheme becomes

$$S(t+dt) = \log(S(t)) + e^{rS(t)dt+\sqrt{\max(\nu(t),0)}S(t)dW^{\mathbb{Q},1}(t)}$$
$$\nu(t+dt) = (e^{-\kappa dt}\nu(t) + (1-e^{-\kappa dt})\theta) \cdot e^{-\frac{1}{2}\Gamma(t)^2dt+\Gamma(t)\cdot dW^{\mathbb{Q},2}(t)}$$

where $\Gamma(t)^2 = \frac{1}{dt}\log\left(1+\frac{\frac{\sigma^2}{2\kappa}\nu(t)(1-e^{-2\kappa dt})}{(e^{-\kappa dt}\nu(t)+(1-e^{-\kappa dt})\theta)^2}\right)$. This scheme produces the following plot

[8] If the reader wishes to calculate these moments they could do so using Exercise 3.1.



Figur 8: Moment matching discretization scheme.

From figure 8 notice how the blue curve seems to give a better fit for higher strike levels compared to the full-truncation scheme (figure 7), however, the plots seems to be very similar in general. We might want to use moment matching anyway as it is a convenient method. We don't need any 'fixes' to prevent the variance from becoming negative. Further, the runtime is 0.67 seconds which is 40% faster compared to using full-truncation. Moment matching is in general a very useful method as it can be applied to an extremely wide set of input distributions and requires a limited number of phases.

Variance reduction by control variate

This method is described in section 2.3 in Broadie, Boyle & Glasserman (1997). Suppose we wish to price a derivative, $Call^{Heston}$, so we simulate N sample paths and compute N potential discounted payoffs, $\widehat{P}_{A,i}$, $i = 1...N$. Thus, the standard Monte Carlo estimate for the price of this call is simply the arithmetic average:

$$P_A = \frac{1}{N}\sum_{i=1}^{N}\widehat{P}_{A,i}$$

The idea in the referenced article is that we use these same N simulated paths to price a derivative, $Call^{BS}$, which is equivalent in every respect except that a geometric average replaces the arithmetic average of the underlying, and use this vanilla option as a control variate in the pricing of the Heston call.

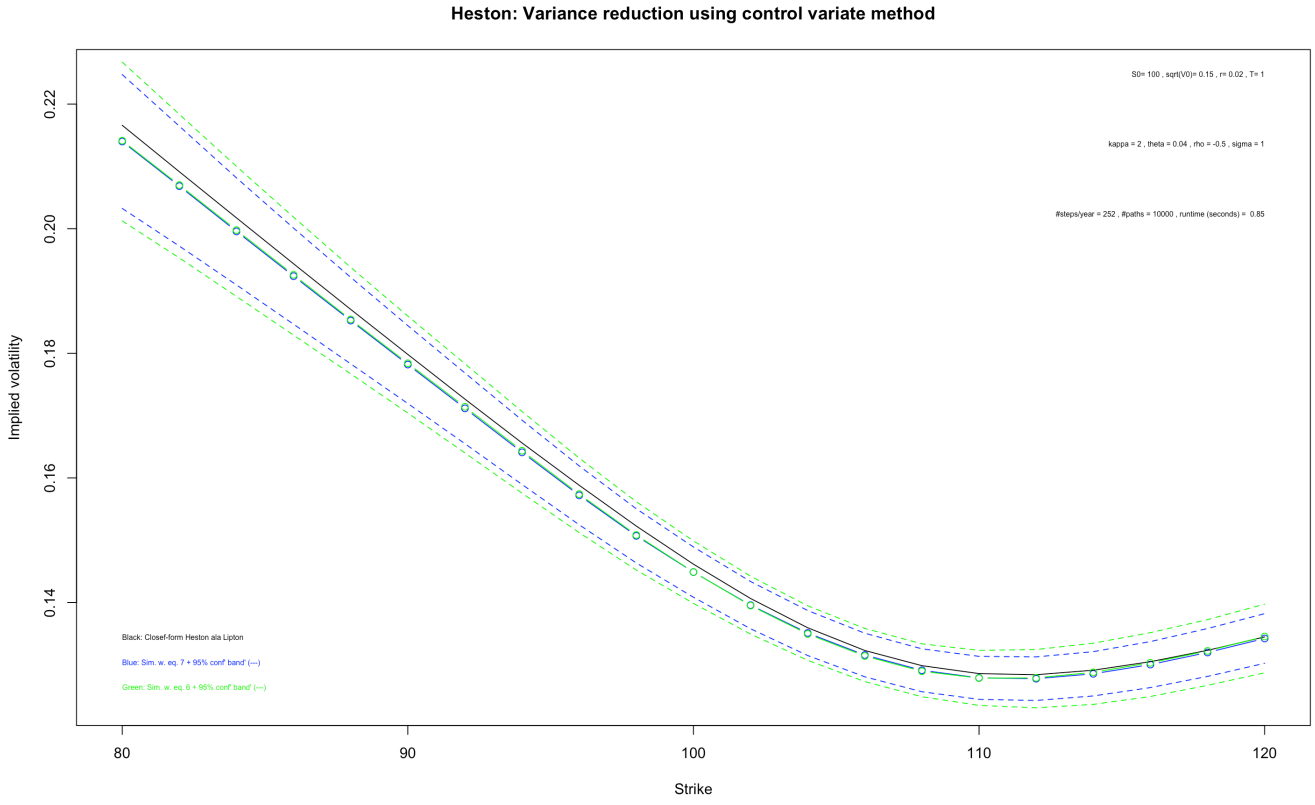
Let P_A denote the average payoff of the Heston call and P_G the average Black-Scholes call payoff. P_G can be evaluated in closed-form and this knowledge can then be used to compute the payoff of the Heston call, P_A. We write $P_A = \mathbb{E}[\widehat{P}_A]$ and $P_G = \mathbb{E}[\widehat{P}_G]$, where $\widehat{P}_A$ and $\widehat{P}_G$ are the discounted option payoffs for one simulated path of the underlying. We then wish to compare the results of using equation (6) and equation (7) from the referenced article:

$$\textcircled{6} \qquad \widehat{P}_A^{cv} = \widehat{P}_A + (P_G - \widehat{P}_G)$$

$$\textcircled{7} \qquad \widehat{P}_A^{\beta} = \widehat{P}_A + \beta(P_G - \widehat{P}_G), \text{ with variance-minimizing } \beta^* = \frac{Cov(\widehat{P}_G, \widehat{P}_A)}{Var(\widehat{P}_G)}$$

The intuition in (6) is that the term $(P_G - \widehat{P}_G)$ acts like a 'control' and adjusts the natural estimator, $\widehat{P}_A$, according to the difference between the average Black-Scholes call payoff value, P_G, and 'the observed value', $\widehat{P}_G$, i.e. the Black-Scholes option value based on one path.

The idea in (7) is then to scale this bias in a way that minimizes the variance of $\widehat{P}_A^{cv}$, thus making the variance of $\widehat{P}_A^{\beta}$ smaller or equal to the variance of $\widehat{P}_A^{cv}$. Using these methods produces the following plot



Figur 9: The green line denotes that eq. (6) has been used in the simulated Heston call price and the blue line denotes that eq. (7) has been used. The black line denotes the true price.

We note that the green and blue dots, i.e. the simulated implied volatilities, match to an extend where it is difficult to tell them apart. Meanwhile, we see that using equation (7) narrows the confidence bands significantly, i.e. reduces the variance, as expected. Comparing to the full-truncation scheme (figure 7) the code runtime is only marginally faster at 0.85 seconds and in general the gains seems to be moderate at best. Why might we want to use the method anyway? Normally when minimizing the error in Monte Carlo simulation, $\frac{\epsilon^2}{n}$, we would increase the number of simulations, n. However, increasing number of simulations also increases runtime. Instead, the idea with variance reduction techniques is to reduce the variance, ϵ^2, thus improving the efficiency of Monte Carlo methods without significantly increasing runtime.

2.b: Hedging

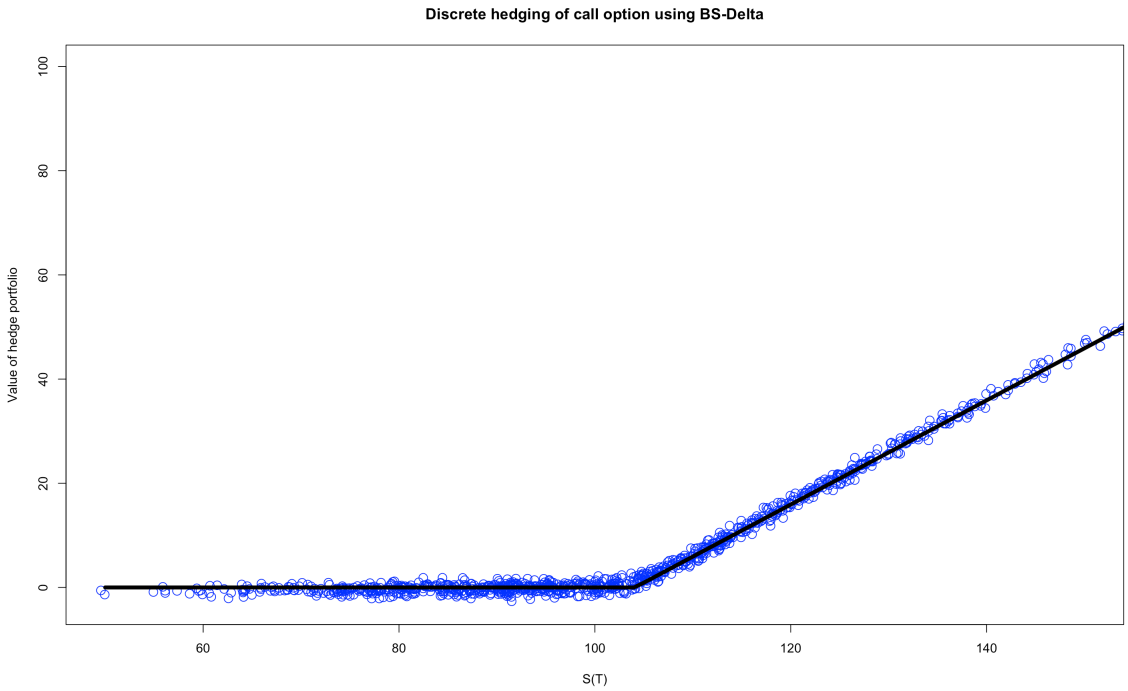
In this problem we wish to perform three different hedge experiments. We will use our usual hedge simulation set-up and the three different ways listed below to try and hedge a call-option by means of the stock and the bank account in the Heston model. When simulating we will use the same simulation parameters as have been used in a previous paper, see Table 2 in Poulsen, Schenk-Hoppe & Ewald (2009).

- Using the Black-Scholes Δ with hedging volatility $\sqrt{\nu(t)}$.
- Using the true Δ of the call-option in the Heston model.
- Using the risk-minimizing Δ.[9]

Black-Scholes Δ

We take a long position in the call option and wish to hedge this with a short position of $\Delta^{BS}(t, S(t), \nu(t))$ units of stock while the bank account is used to make the portfolio self-financing. The code can be found in the Appendix and the result is seen in figure 10.

[9] Equation (6) in Poulsen, Schenk-Hoppe & Ewald (2009).



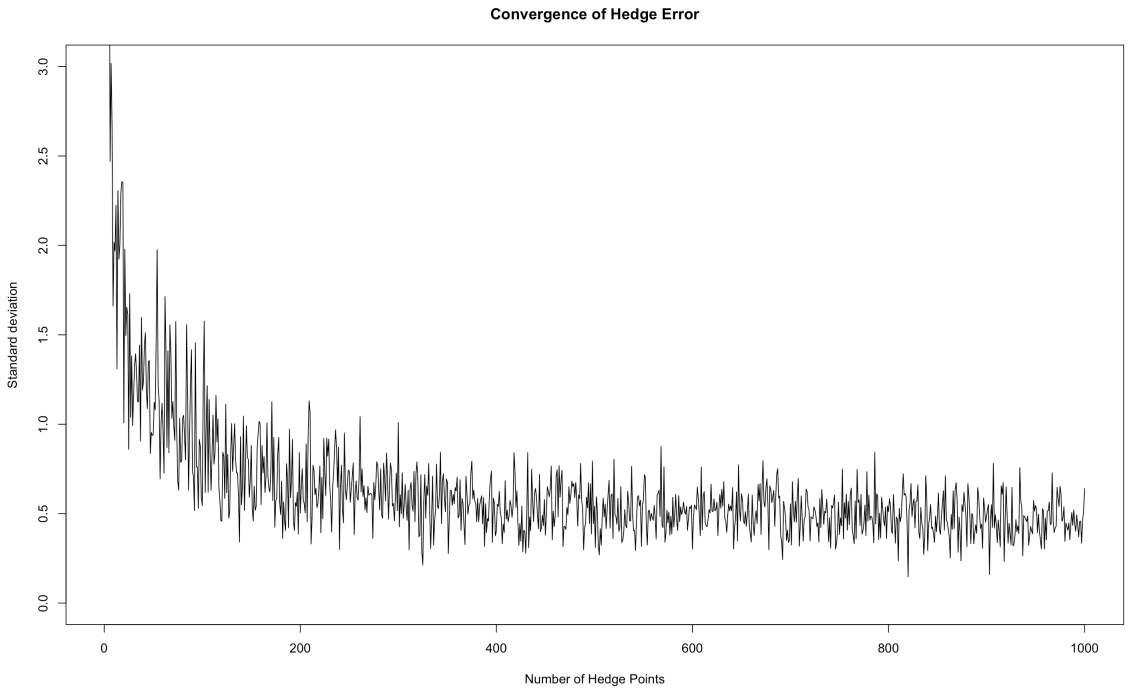
Figur 10: Discrete hedge of call option in the Heston model using Δ^{BS} and the following parameters: $S(0) = 100$, $T = 1$, $r = 0.04$, $\theta = 0.22^2$, $\kappa = 4.75$, $\sigma = 0.1$, $\rho = -0.5$, $V(0) = \theta$, $K = S(0)e^{rT}$ and 1000 paths with 252 hedge points a year.

This looks as if it's a decent hedge, but is it perfect? We examine the mean and standard deviation of the hedge error.[10] If the hedge is perfect we will expect to see a mean with a value of zero and a smooth convergence of the standard deviation towards zero.

Hedges pr. year	Standard Dev.	Mean
252	0.6995	0.0488
500	0.5668	−0.0276
1000	0.4462	0.0795
2000	0.4703	−0.0987

Tabel 1: Standard deviation and mean of hedge error, of strategy using Δ^{BS} for different hedge points pr. year based on 100 simulations.

[10]The hedge error is the difference between the portfolio value and the option payoff, of which we wish to hedge.



Figur 11: Convergence of standard deviation of hedge error of a discrete hedge of call option in the Heston model using Δ^{BS}.

From table 1 we note that neither the mean nor the standard deviation comes close enough to zero when headges per year increases. Further, it can be seen in Figure 11 that the convergence of the standard deviation is far from smooth. Thus, the hedging strategy does not work perfectly. However this does not surprise us, as the Heston model is incomplete due to the presence of volatility risk represented by the $dW^{\mathbb{Q},2}(t)$-term. This is also why above result does not contradict the Fundamental Theorem of Derivative Trading. As stated in Poulsen (2018)[11] - our formula for the Fundamental Theorem of Derivaive Trading is based on the Black-Scholes assumption of constant volatility. In order to say anything valuable with FTODT in the Heston model we would first need to derive this under the stochastic volatility assumption. Thus, where the generalized Black-Scholes model is complete and we can eliminate all risk, this is not possible in the Heston model using only the bank account and underlying asset.

[11] Page 2, Paragraph *"Stochastic Volatility"*

Heston Δ

We now wish to investigate whether hedging using the true Δ of the call-option in the Heston model gives a perfect hedge. We will first need to derive it. We know that the call option price in the Heston model is

$$Call(t, S(t), \nu(t)) = S(t)\mathbb{Q}_t^S(S(T) \geq K) - Ke^{-r(T-t)}\mathbb{Q}_t(S(T) \geq K)$$

For $\lambda > 0$ consider a call with strike λK and underlying stock process $(\lambda S(t))$. Then the price of this call is

$$Call(t, \lambda S(t), \nu(t)) = \lambda S(t)\mathbb{Q}_t^S(S(T) \geq K) - \lambda Ke^{-r(T-t)}\mathbb{Q}_t(S(T) \geq K) = \lambda Call(t, S(t), \nu(t))$$

This means that the call price is homogeneous of degree one w.r.t. spot, $S(t)$, and strike, K. Therefore we know by Eulers homogeneity theorem that

$$\begin{aligned} Call(t, S(t), \nu(t)) &= S(t)Call_S(t, S(t), \nu(t)) + KCall_K(t, S(t), \nu(t)) \\ \Leftrightarrow \Delta^{Heston}(t) = Call_S(t, S(t), \nu(t)) &= \frac{Call(t, S(t), \nu(t)) - KCall_K(t, S(t), \nu(t))}{S(t)} \end{aligned}$$

What remains is then to determine the derivative of the call price w.r.t. strike. First, by risk neutral valuation

$$Call_K(t, S(t), \nu(t)) = \frac{\partial}{\partial K}\left(e^{-r(T-t)}\mathbb{E}_t^{\mathbb{Q}}\left[(S(T) - K)^+\right]\right)$$

Let φ be the conditional density of $S(T)$ w.r.t. the Lebesgue measure on $\mathbb{R}$ under $\mathbb{Q}$. Then the expectation can be written as

$$= \frac{\partial}{\partial K}\left(e^{-r(T-t)}\int_K^\infty (y - K)\varphi(y)dy\right)$$

Using Leibniz's Integral Rule, assuming that the density goes to zero so rapidly that it 'kills off' everything, yields

$$= 0 + e^{-r(T-t)}\int_K^\infty \frac{\partial}{\partial K}(y - K)\varphi(y)dy$$

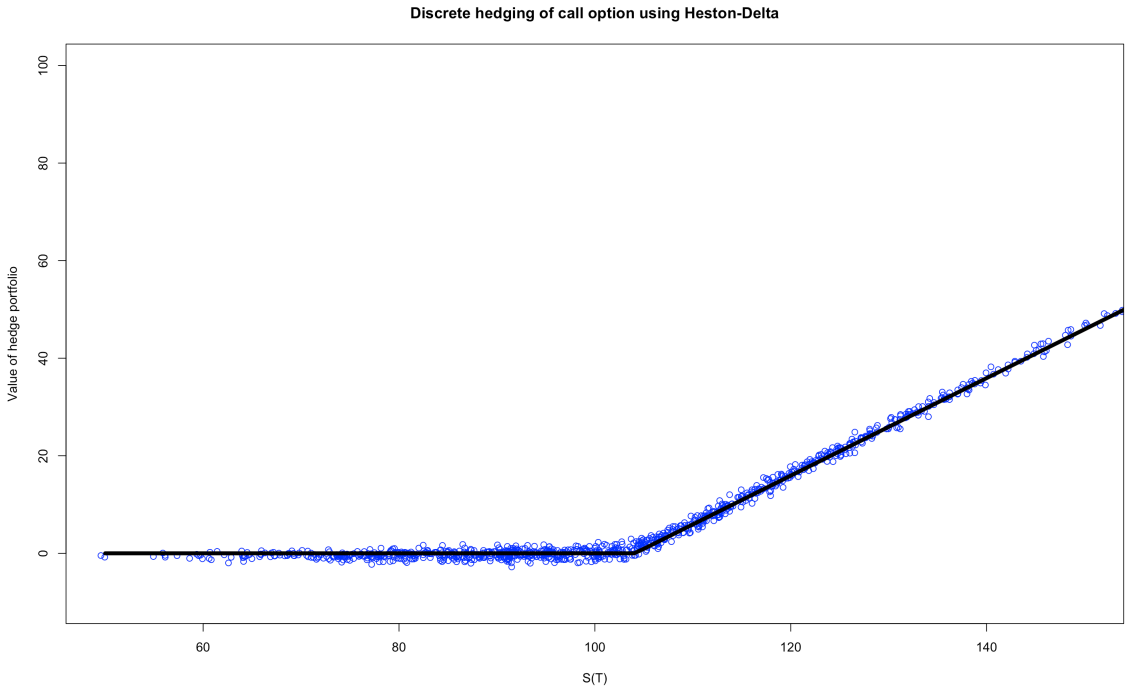
We have not specified an explicit form of $\varphi(y)$, but all we need to know is that it does not depend on K.

$$\begin{aligned}
&= e^{-r(T-t)} \int_K^\infty (-1) \cdot \varphi(y) dy \\
&= -e^{-r(T-t)} \mathbb{E}_t^{\mathbb{Q}}[\mathbb{1}_{S(T) \geq K}] \\
&= -e^{-r(T-t)} \mathbb{Q}_t(S(T) \geq K)
\end{aligned}$$

Combining the above we find the true Δ of the call option in the Heston model to be

$$\begin{aligned}
\Delta^{Heston}(t) &= \frac{1}{S(t)} \Big(S(t) \mathbb{Q}_t^S(S(T) \geq K) - K e^{-r(T-t)} \mathbb{Q}_t(S(T) \geq K) - K(-e^{-r(T-t)} \mathbb{Q}_t(S(T) \geq K)) \Big) \\
&= \mathbb{Q}_t^S(S(T) \geq K)
\end{aligned}$$

The intuition here is that we should invest an amount equal to the $\mathbb{Q}^S$-probability of the call option ending up ITM, in the stock. We take a long position in the call option and wish to hedge this with a short position of Δ^{Heston} units of stock while the bank account is used to make the portfolio self-financing. The code can be found in the Appendix and the result is seen in figure 12.

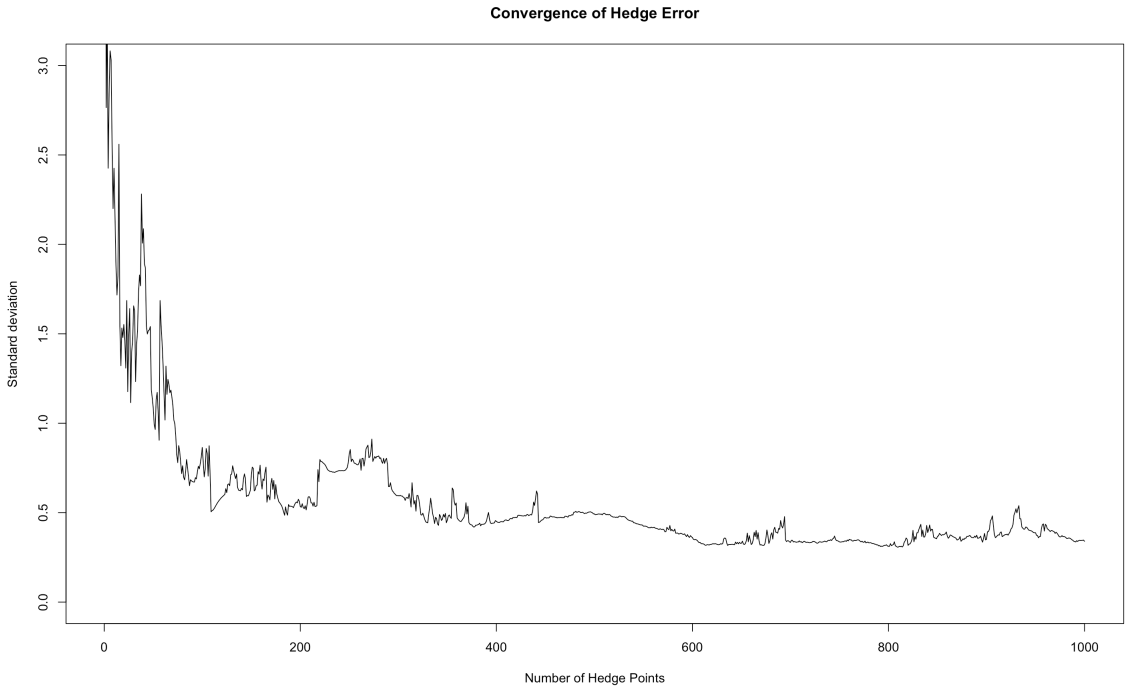


Figur 12: Discrete hedge of call option in the Heston model using Δ^{Heston} and the following parameters: $S(0) = 100$, $T = 1$, $r = 0.04$, $\theta = 0.22^2$, $\kappa = 4.75$, $\sigma = 0.1$, $\rho = -0.5$, $V(0) = \theta$, $K = S(0)e^{rT}$ and 1000 paths with 252 hedge points a year.

To evaluate the hedge we once again take a look at the mean and standard deviation of the hedge error.

Hedges pr. year	Standard Dev.	Mean
252	0.6068	0.0086
500	0.5642	−0.0418
1000	0.4934	−0.0294
2000	0.4419	0.0009

Tabel 2: Standard deviation and mean of hedge error, of strategy using Δ^{Heston} for different hedge points pr. year based on 100 simulations.



Figur 13: Convergence of standard deviation of hedge error of a discrete hedge of call option in the Heston model using the true Δ^{Heston}.

Looking at table 2 we notice how the value of the mean is still not zero and looking at figure 13 that the standard deviation does not converge smoothly towards zero either. Thus, the hedging strategy is still not perfect. However, this is not a surprise to us as the market is incomplete. One could though note that comparing hedge error mean and smoothness to the previous bulletpoint (table 1 and figure 11), where a Black-Scholes delta was used to hedge, this strategy seems to perform much better.

Risk-minimizing Δ

We now wish to investigate whether hedging using a risk-minimizing Δ provides a perfect hedge in the Heston model. A derivation of such is made following section 2.4 in Poulsen, Schenk-Hoppe & Ewald (2009). We set up a hedging portfolio consisting of one long European option and short h_S units of stock. By Ito's formula we find the option dynamics, where the dt-term does not matter in the analysis of conditional variance,

$$dC(t, S(t), \nu(t)) = (...)dt + C_S dS(t) + C_\nu d\nu(t)$$

Thus, the value of the hedging portfolio, i.e. $\pi(t) = C(t, S(t), \nu(t)) - h_S S(t)$, has dynamics given by[12]

$$d\pi(t) = 1 \cdot dC(t, S(t), \nu(t)) - h_S \cdot dS(t) = (...)dt + (C_S - h_S)dS(t) + C_\nu d\nu(t)$$

The locally risk minimizing hedge, Δ^{min}, is then the value of h_S which minimizes the conditional variance of changes in the hedge portfolio value.[13] We find

$$Var_t[d\pi(t)] = (C_S - h_S)^2 Var_t[dS(t)] + C_\nu^2 Var_t[d\nu(t)] + 2(C_S - h_S)C_\nu Cov_t[dS(t), d\nu(t)]$$

We note that, as $(dt)^2 = 0$, $\mathbb{E}[dW^{\mathbb{Q},1}(t)] = \mathbb{E}[dW^{\mathbb{Q},2}(t)] = 0$ and $\mathbb{E}[dW^{\mathbb{Q},1}(t)dW^{\mathbb{Q},1}(t)] = \rho dt$,

$$Var_t[dS(t)] = \mathbb{E}_t[(dS(t))^2] - \underbrace{\Big(\mathbb{E}_t[dS(t)]\Big)^2}_{=0} = \nu(t)S(t)dt$$

$$Var_t[d\nu(t)] = \mathbb{E}_t[(d\nu(t))^2] - \underbrace{\Big(\mathbb{E}_t[d\nu(t)]\Big)^2}_{=0} = \sigma^2\nu(t)dt$$

$$Cov_t[dS(t), d\nu(t)] = \mathbb{E}_t[dS(t)d\nu(t)] - \underbrace{\mathbb{E}_t[dS(t)]\mathbb{E}_t[d\nu(t)]}_{=0} = \nu(t)S(t)\sigma\rho dt$$

Then

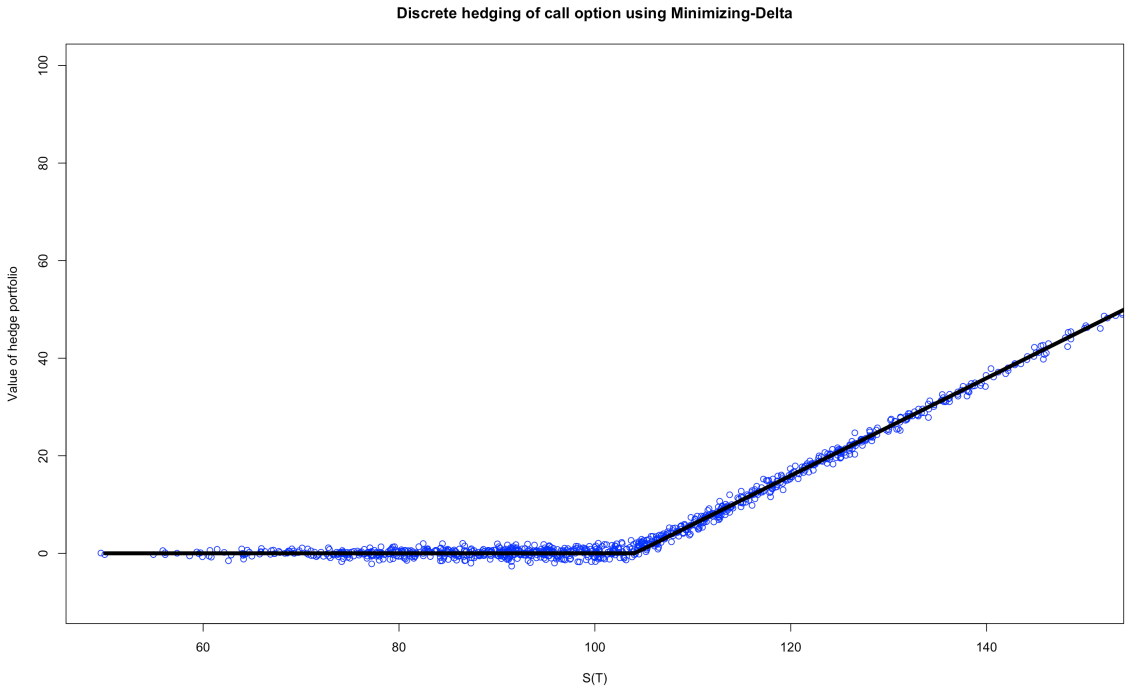
$$Var_t[d\pi(t)] = (C_S - h_S)^2\nu(t)S(t)dt + C_\nu^2\sigma^2\nu(t)dt + 2(C_S - h_S)C_\nu\nu(t)S(t)\sigma\rho dt$$

[12] By self-financing property: Björk Definition 6.2
[13] Such value can be found as the function $Var_t[d\pi(t)]$ is convex in h_S.

Thus the first order condition is

$$
\begin{aligned}
0 = \frac{\partial Var_t[d\pi(t)]}{\partial h_S} &= 2\left(S(t)^2\nu(t)(h_S - C_S) - C_\nu S(t)\nu(t)\sigma\rho\right)dt \\
\Leftrightarrow h_S &= C_S + \frac{\rho\sigma C_\nu}{S(t)} \\
\Leftrightarrow \Delta^{min} &= \Delta^{Heston} + \frac{\rho\sigma C_\nu}{S(t)}
\end{aligned}
$$

This is exactly equation (6) in the linked article. As stated in Poulsen, Schenk-Hoppe & Ewald (2009) this formula, and thereby our hedge, depends on the pricing function C, which depends on a non-defined martingale measure. When simulating we will need to make some assumptions about this martingale measure. We will implement the strategy in R using the same call option formula as in the two previous strategies. The code can be found in the Appendix and the result is seen in figure 14

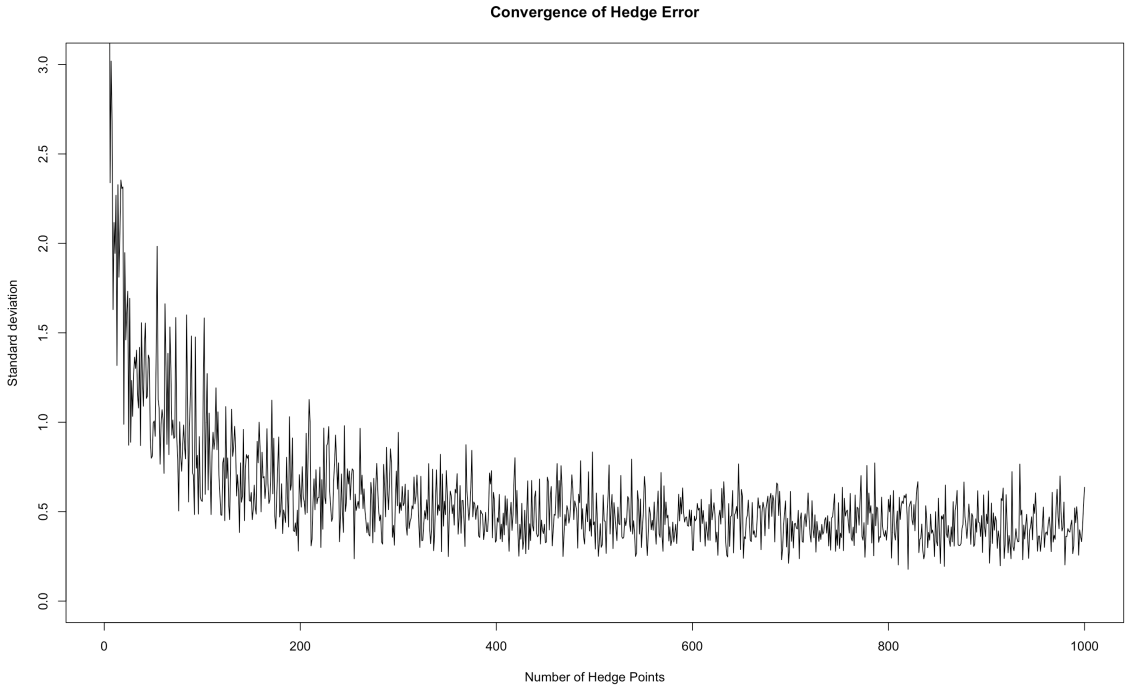


Figur 14: Discrete hedge of call option in the Heston model using Δ^{min} and the following parameters: $S(0) = 100$, $T = 1$, $r = 0.04$, $\theta = 0.22^2$, $\kappa = 4.75$, $\sigma = 0.1$, $\rho = -0.5$, $V(0) = \theta$, $K = S(0)e^{rT}$ and 1000 paths with 252 hedge points a year.

This looks like a decent hedge, so we again examine the hedge error.

Hedges pr. year	Standard Dev.	Mean
252	0.6605	0.0559
500	0.5261	−0.013
1000	0.4291	0.059
2000	0.4256	−0.0524

Tabel 3: Standard deviation and mean of hedge error, of strategy using Δ^{min}, for different hedge points pr. year based on 100 simulations.



Figur 15: Convergence of standard deviation of hedge error of a discrete hedge of call option in the Heston model using the true Δ^{min}.

We note from table 3 that the value of the mean is still not as close to 0 as one might want it to be and from figure 15 that the standard deviation does not converge smoothly towards zero, meaning this hedge is not perfect either. These conclusions are not surprising though as the model is incomplete. However, we notice that, comparing hedge error mean and smoothness, this strategy seems to perform worse than the previous (table 2 and figure 13), where the true Heston delta was used to hedge. This could be an indication that the martingale measure is not optimal, and the hedge could be improved if such were specified.

3 Appendix

1.b

```
# Functions
BlackScholesFormula <- function(spot,strike,timetomat,r,q=0,sigma,option){

  d <- function(spot,strike,r,q,sigma,timetomat,type){

    if (type == 1) result <- (log(spot/strike)+(r-q)*timetomat +
    (1/2)*(sigma^2)*timetomat)/(sigma*sqrt(timetomat))

    if (type == 2) result <- (log(spot/strike)+(r-q)*timetomat -
    (1/2)*(sigma^2)*timetomat)/(sigma*sqrt(timetomat))

    d <- result
  }

  d1 <- d(spot,strike,r,q,sigma,timetomat,type=1)
  d2 <- d(spot,strike,r,q,sigma,timetomat,type=2)

  if (option=="Call"){result<-exp(-q*timetomat)*spot*pnorm(d1) -
  strike*exp(-r*timetomat)*pnorm(d2)}
  if (option=="Put"){result<- exp(-q*timetomat)*spot*pnorm(d1) -
  strike*exp(-r*timetomat)*pnorm(d2) -exp(-q*timetomat)*spot + strike*exp(-r*timetomat)}

  BlackScholesFormula <- result

}

BlackScholesImpVol <-function (obsprice, spot, timetomat, strike, r, q=0, option= "Call") {
difference <- function(sigBS, obsprice, spot, timetomat, strike, r, q,option) {
BlackScholesFormula(spot, strike,timetomat, r, q=0, sigBS,option) - obsprice}

uniroot(difference, c(10^-6,10), obsprice = obsprice,
        spot = spot, timetomat = timetomat, strike = strike,
```

```
            r = r, q = q, option = option)$root
}
BlackScholesImpVol <- Vectorize(BlackScholesImpVol)

price_Bach <- function(spot, timetomat, strike, sigma){
  d <- (spot - strike)/(sigma * sqrt(timetomat))
  price_Bach <- (spot - strike)*pnorm(d) + sigma*sqrt(timetomat)*dnorm(d)
}
price_Bach <- Vectorize(price_Bach)

# Parameters
S0 <- 100
T <- 1
sigma <- 15
r <- 0
strike <- 1:300

# Plot
prices <- price_Bach(S0, T, strike, sigma)
imp_vols <- BlackScholesImpVol(prices, S0, T, strike, r)

plot(strike, imp_vols, col="blue", type = 'l',
     xlab = "Strike", ylab = "Implied volatilities",
     main = "Implied Volatility Skew in the Bachelier model")
```

1.d

```
S0 <- 100
T <- 1
sigma <- 15
K <- c(20:200)
t <- 0
S <- S0

Call_Bach <- function(S,t,T,sigma,K){
  (S-K)*pnorm((S-K)/(sigma*sqrt(T-t)))+sigma*sqrt(T-t)*dnorm((S-K)/(sigma*sqrt(T-t)))
```

```
}
Call_Bach <- Vectorize(Call_bach)

C_T <- function(eps,T){
  f1 <- Call_Bach(S0,0,T+eps,sigma,K)
  f2 <- Call_Bach(S0,0,T,sigma,K)
  (f1-f2)/eps
}

C_KK <- function(eps,K){
  f1 <- Call_Bach(S0,0,T,sigma,K)
  f2 <- Call_Bach(S0,0,T,sigma,K+eps)
  f3 <- Call_Bach(S0,0,T,sigma,K-eps)
  (f2-2*f1+f3)/(eps^2)
}

Dupire <- function(K,CapT,epsK,epsT){
  C_T(epsT,T)/((1/2)*K^2*C_KK(epsK,K))
}

par(mfrow=c(2,3))
n <- 2:7
for(i in n){
  V <- Dupire(K,T,epsT =10^{-i}, epsK=10^{-i})
  plot(K,V, main=bquote("epsilon = " ~ 10**- .(i)),ylab = "Local Volatility", xlab = "Strike")
  points(K,sigma^2/(K^2), type = 'l', col = 'red')
}
```

2.a

```
source("BlackScholesFormula.R")
source("Fourier.R")

timetoexp<-1.0
S0<-100
R<-0.02
```

```
V0<-0.15^2
kappa<-2
theta<-0.2^2
sigma<-1.0
rho<--0.5

Params<-paste("S0=", S0, ", sqrt(V0)=",sqrt(V0),", r=",R, ", T=", timetoexp)
ModelParams<-paste("kappa =", kappa, ", theta =", theta, ", rho =", rho, ", sigma =", sigma)

Nsim<-10^4
NstepsPerYear<-1*252
Nsteps<-round(timetoexp*NstepsPerYear)
dt<-timetoexp/Nsteps

S_BS <- rep(S0,Nsim); V<-rep(V0,Nsim); S <- rep(S0,Nsim)

### Change to TRUE to see repective plot
AbsAtZero<-FALSE
if (AbsAtZero) title<-"Heston: Euler-S + |V| @0"
MaxAtZero<-FALSE
if (MaxAtZero) title<-"Heston: Euler-S + V^+ @0"
FullTruncation<-FALSE
if (FullTruncation) title<-"Heston\n Euler-ln(S) + full truncation-V"
MomentMatching<-FALSE
if (MomentMatching) title<-"Heston: Euler-ln(S) + moment matching-V"
VarianceReduction<-TRUE
if (VarianceReduction) title<- "Heston: Variance reduction using control variate method"

IVspace<-TRUE

set.seed(24)
RunTime<-system.time(
for (i in 1:Nsteps){
  dW1<-sqrt(dt)*rnorm(Nsim,0,1)
  dW2<-rho*dW1+sqrt(1-rho^2)*sqrt(dt)*rnorm(Nsim,0,1)
  if (AbsAtZero){
```

```
    S<-S+R*S*dt+sqrt(V)*S*dW1
    V<-abs(V + kappa*(theta-V)*dt + sigma*sqrt(V)*dW2)
  }
  if (MaxAtZero){
    S<-S+R*S*dt+sqrt(V)*S*dW1
    V<-pmax(V + kappa*(theta-V)*dt + sigma*sqrt(V)*dW2,0)
  }

  if (FullTruncation){
    S<-exp(log(S)+(R-0.5*pmax(V,0))*dt+sqrt(pmax(V,0))*dW1)
    V<-V+kappa*(theta-pmax(V,0))*dt+sigma*sqrt(pmax(V,0))*dW2
  }
  if (MomentMatching){
  S<-exp(log(S)+(R-0.5*V)*dt+sqrt(V)*dW1)
  Gammasquared <-(1/dt)*log(1+((1/2)*(sigma^2)*(1/kappa)*V*(1-exp(-2*kappa*dt)))
    /((exp(-kappa*dt)*V+(1-exp(-kappa*dt))*theta)^2))
  V <- (exp(-kappa*dt)*V+(1-exp(-kappa*dt))*theta)*exp((-1/2)*Gammasquared*dt+sqrt(Gammasquared)*dW2)
  }
  if (VarianceReduction){
    S_BS <- exp(log(S_BS)+(R-0.5*V0)*dt+sqrt(V0)*dW1)
    S<-exp(log(S)+(R-0.5*pmax(V,0))*dt+sqrt(pmax(V,0))*dW1)
    V<-V+kappa*(theta-pmax(V,0))*dt+sigma*sqrt(pmax(V,0))*dW2
  }
}
)[3]

ExperimentParams<-paste("#steps/year =",NstepsPerYear, ", #paths =", Nsim, ",
    runtime (seconds) = ", round(RunTime,2))

StdErrSimCall<-StdErrSimCall_cv<-TrueCall<-SimCall<-SimCall_cv<-Strikes<-S0+(-10:10)*2
ConfBandIVSimCall<-ConfBandIVSimCall_cv<-IVSimCall<-IVSimCall_cv<-IVTrueCall<-SimCall

if (!VarianceReduction){
  for (i in 1:length(SimCall)) {
    SimCall[i]<-exp(-R*timetoexp)*mean(pmax(S-Strikes[i],0))
    StdErrSimCall[i]<-sd(exp(-R*timetoexp)*(pmax(S-Strikes[i],0)))/sqrt(Nsim)
```

```
    if (IVspace) IVSimCall[i]<-BlackScholesImpVol(SimCall[i],S0,timetoexp,Strikes[i],
        R, q=0, opttype=1)
    if (IVspace) ConfBandIVSimCall[i]<-BlackScholesImpVol(SimCall[i]+1.96*StdErrSimCall[i],S0,
        timetoexp,Strikes[i],R, q=0, opttype=1)
    TrueCall[i]<-Heston.Fourier(S0,timetoexp,Strikes[i],R,0,V0,theta,kappa,sigma,rho,greek=1)
    if (IVspace) IVTrueCall[i]<-BlackScholesImpVol(TrueCall[i],S0,timetoexp,Strikes[i],
        R, q=0, opttype=1)
  }
}

if (VarianceReduction){
  for (i in 1:length(SimCall)) {
    hatP_H <- exp(-R*timetoexp)*pmax(S-Strikes[i],0)
    hatP_BS <- exp(-R*timetoexp)*pmax(S_BS-Strikes[i],0)
    P_CV <- hatP_H + (BlackScholesFormula(S0,timetoexp,Strikes[i],R, q=0, sqrt(V0),
        opttype=1, greektype=1) - hatP_BS)
    beta <- cov(hatP_H, hatP_BS) / var(hatP_BS)
    P_beta <- hatP_H + beta*(BlackScholesFormula(S0,timetoexp,Strikes[i],R, q=0,
        sqrt(V0), opttype=1, greektype=1) - hatP_BS)
    SimCall[i]<- mean(P_beta)
    StdErrSimCall[i]<- sd(P_beta)/sqrt(Nsim)
    if (IVspace) IVSimCall[i]<-BlackScholesImpVol(SimCall[i],S0,timetoexp,Strikes[i],
        R, q=0, opttype=1)
    if (IVspace) ConfBandIVSimCall[i]<-BlackScholesImpVol(SimCall[i]+1.96*StdErrSimCall[i],
        S0,timetoexp,Strikes[i],R, q=0, opttype=1)
    TrueCall[i]<-Heston.Fourier(S0,timetoexp,Strikes[i],R,0,V0,theta,kappa,sigma,rho,greek=1)
    if (IVspace) IVTrueCall[i]<-BlackScholesImpVol(TrueCall[i],S0,timetoexp,Strikes[i],
        R, q=0, opttype=1)

    SimCall_cv[i]<- mean(P_CV)
    StdErrSimCall_cv[i]<- sd(P_CV)/sqrt(Nsim)
    if (IVspace) IVSimCall_cv[i]<-BlackScholesImpVol(SimCall_cv[i],S0,timetoexp,Strikes[i],
        R, q=0, opttype=1)
    if (IVspace) ConfBandIVSimCall_cv[i]<-BlackScholesImpVol(SimCall_cv[i]+1.96*StdErrSimCall_cv[i],
        S0,timetoexp,Strikes[i],R, q=0, opttype=1)
  }
```

```
}

if(IVspace){
  dummy<-c(min(2*IVSimCall-ConfBandIVSimCall,IVTrueCall),max(IVTrueCall,ConfBandIVSimCall) )
  plot(Strikes,IVSimCall,type='b',col='blue',ylim=dummy, ylab="Implied volatility",
       main=title,xlab="Strike")
  points(Strikes,IVTrueCall,type='l')
  points(Strikes,ConfBandIVSimCall,type='l',lty=2,col='blue')
  points(Strikes,2*IVSimCall-ConfBandIVSimCall,type='l',lty=2,col='blue')
  if (VarianceReduction) points(Strikes,IVSimCall_cv,col='green')
  if (VarianceReduction) points(Strikes,ConfBandIVSimCall_cv,type='l',lty=2,col='green')
  if (VarianceReduction) points(Strikes,2*IVSimCall-ConfBandIVSimCall_cv,type='l',lty=2,col='green')

  text(min(Strikes),min(dummy)+0.10*(max(dummy)-min(dummy)),
    "Black: Closef-form Heston ala Lipton",adj=0,cex=0.5)
  text(min(Strikes),min(dummy)+0.06*(max(dummy)-min(dummy)),
    "Blue: Sim. w. eq. 7 + 95% conf' band' (---)", col="blue",adj=0,cex=0.5)
  if (VarianceReduction) text(min(Strikes),min(dummy)+0.02*(max(dummy)-min(dummy)),
    "Green: Sim. w. eq. 6 + 95% conf' band' (---)", col="green",adj=0,cex=0.5)

  text(max(Strikes),max(dummy),Params,adj=1, cex=0.5)
  text(max(Strikes),0.95*max(dummy),ModelParams,adj=1, cex=0.5)
  text(max(Strikes),0.90*max(dummy),ExperimentParams,adj=1, cex=0.5)
}
```

2.b

Discrete hedging of call option in the Heston model using BS-Delta

```
source("BlackScholesFormula.R")
source("Fourier.R")

# Parameters
r <- 0.04
theta <- 0.22^2
kappa <- 4.75
sigma <- 0.1
```

```
rho <- -0.5
S0 <- 100
V0 <- theta
T <- 1
K <- S0*exp(r*T)

nHedge <- 252*T
nPath <- 1000
St <-rep(S0, length = nPath)
Vt <- rep(V0, length(nPath))
dt <- T/nHedge

initialoutlay <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = r,rho,greek=1)
pfValue <-rep(initialoutlay, length = nPath)
a <- BlackScholesFormula(St, T, K, r, 0, sqrt(Vt), opttype = 1, greektype = 2)
b <- pfValue - a*St

# Simulation
set.seed(24)

for (i in 2:nHedge){
dW1<-sqrt(dt)*rnorm(nPath,0,1)
dW2<-rho*dW1+sqrt(1-rho^2)*sqrt(dt)*rnorm(nPath,0,1)
St<-St*exp((r-0.5*pmax(Vt,0))*dt+sqrt(pmax(Vt,0))*dW1)
Vt<-Vt+kappa*(theta-pmax(Vt,0))*dt+sigma*sqrt(pmax(Vt,0))*dW2
pfValue <- a*St+b*exp(r*dt)
a <- BlackScholesFormula(St, T-(i-1)*dt, K, r, 0, sqrt(Vt), opttype = 1, greektype = 2)
b <- pfValue - a*St
}

plot(St, pfValue, col="blue", xlab ="S(T)", ylab =" Value of hedge portfolio ", ylim=c(-3, 100),
xlim=c(50, 150), cex=1.5, main = "Discrete hedging of call option using BS-Delta")
points(50:200, (50:200 - K)*(50:200 >= K), type ='l', lwd =5)

## Hedge Experiment
```

```
HedgeExperiment <- function(nHedge, nPath){

  St <-rep(S0, length = nPath)
  Vt <- rep(V0, length(nPath))
  dt <- T/nHedge

  initialoutlay <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = sigma,rho,greek=1)
  pfValue <-rep(initialoutlay, length = nPath)
  a <- BlackScholesFormula(St, T, K, r, 0, sqrt(pmax(Vt,0)), opttype = 1, greektype = 2)
  b <- pfValue - a*St

  for (i in 2:nHedge){
    dW1<-sqrt(dt)*rnorm(nPath,0,1)
    dW2<-rho*dW1+sqrt(1-rho^2)*sqrt(dt)*rnorm(nPath,0,1)
    St<-St*exp((r-0.5*pmax(Vt,0))*dt+sqrt(pmax(Vt,0))*dW1)
    Vt<-Vt+kappa*(theta-pmax(Vt,0))*dt+sigma*sqrt(pmax(Vt,0))*dW2
    pfValue <- a*St+b*exp(r*dt)
    a <- BlackScholesFormula(St, T-(i-1)*dt, K, r, 0, sqrt(pmax(Vt,0)), opttype = 1, greektype = 2)
    b <- pfValue - a*St
  }

  Payoff <- (St - K)*(St >= K)
  hedgeError <- pfValue - Payoff

  output = list( sd = sd(hedgeError), mean = mean(hedgeError) )
}



# table
(HedgeExperiment_table <- data.frame(c(252, 500, 1000,2000),
                                     rbind(HedgeExperiment(252*T, 100),
                                           HedgeExperiment(500*T, 100),
                                           HedgeExperiment(1000*T, 100),
                                           HedgeExperiment(2000*T, 100))))
# plot
numberOfPaths <- 10
```

```
testHedgePts <- NA
for (j in 1:1000){testHedgePts[j] <- j}
se <- numeric(length(testHedgePts))

for (i in 1:length(testHedgePts)){se[i] <- HedgeExperiment(testHedgePts[i], numberOfPaths)$sd}

plot(testHedgePts, se, xlab = "Number of Hedge Points", ylim=c(0,3),
     type="l", ylab = "Standard deviation", main = "Convergence of Hedge Error")
```

Discrete hedging of call option in the Heston model using Heston-Delta

```
source("BlackScholesFormula.R")
source("Fourier.R")

# Parameters
r <- 0.04
theta <- 0.22^2
kappa <- 4.75
sigma <- 0.1
rho <- -0.5
S0 <- 100
V0 <- theta
T <- 1
K <- S0*exp(r*T)

nHedge <- 252*T
nPath <- 1000
St <-rep(S0, length = nPath)
Vt <- rep(V0, length(nPath))
dt <- T/nHedge

# Hedge
initialoutlay <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = sigma,rho,greek=1)
pfValue <-rep(initialoutlay, length = nPath)

a <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = sigma,rho,greek=2)
```

```
b <- pfValue - a*St

set.seed(24)
for (i in 2:nHedge){
dW1<-sqrt(dt)*rnorm(nPath,0,1)
dW2<-rho*dW1+sqrt(1-rho^2)*sqrt(dt)*rnorm(nPath,0,1)
St<-St*exp((r-0.5*pmax(Vt,0))*dt+sqrt(pmax(Vt,0))*dW1)
Vt<-Vt+kappa*(theta-pmax(Vt,0))*dt+sigma*sqrt(pmax(Vt,0))*dW2
pfValue <- a*St+b*exp(r*dt)
a <- mapply(FUN=Heston.Fourier, spot=St, V =Vt, MoreArgs = list(timetoexp=T-(i-1)*dt,
    strike=K,r=r,divyield=0,theta=theta,kappa=kappa,epsilon = sigma,rho=rho,greek=2))
b <- pfValue - a*St
}

plot(St, pfValue, col="blue", xlab ="S(T)",
    ylab =" Value of hedge portfolio ", ylim=c(-10, 100), xlim=c(50, 150),
    main = "Discrete hedging of call option using Heston-Delta")
points(50:200, (50:200 - K)*(50:200 >= K), type ='l', lwd =5)

## Hedge Error
HedgeExperiment <- function(nHedge, nPath){

  St <-rep(S0, length = nPath)
  Vt <- rep(V0, length(nPath))
  dt <- T/nHedge

  initialoutlay <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = sigma,rho,greek=1)
  pfValue <-rep(initialoutlay, length = nPath)

  a <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = sigma,rho,greek=2)
  b <- pfValue - a*St

  set.seed(24)
  for (i in 2:nHedge){
    dW1<-sqrt(dt)*rnorm(nPath,0,1)
    dW2<-rho*dW1+sqrt(1-rho^2)*sqrt(dt)*rnorm(nPath,0,1)
```

```
    St<-St*exp((r-0.5*pmax(Vt,0))*dt+sqrt(pmax(Vt,0))*dW1)
    Vt<-Vt+kappa*(theta-pmax(Vt,0))*dt+sigma*sqrt(pmax(Vt,0))*dW2
    pfValue <- a*St+b*exp(r*dt)
    a <- mapply(FUN=Heston.Fourier, spot=St, V =Vt, MoreArgs = list(timetoexp=T-(i-1)*dt,
    strike=K,r=r,divyield=0,theta=theta,kappa=kappa,epsilon = sigma,rho=rho,greek=2))
    b <- pfValue - a*St
  }

  Payoff <- (St - K)*(St >= K)
  hedgeError <- pfValue - Payoff

  output = list( sd = sd(hedgeError), mean = mean(hedgeError) )
}



(HedgeExperiment_table <- data.frame(c(252, 500, 1000,2000),
                                     rbind(HedgeExperiment(252*T, 100),
                                           HedgeExperiment(500*T, 100),
                                           HedgeExperiment(1000*T, 100),
                                           HedgeExperiment(2000*T, 100))))
# plot
numberOfPaths <- 10
testHedgePts <- NA
for (j in 1:1000){testHedgePts[j] <- j}
se <- numeric(length(testHedgePts))

for (i in 1:length(testHedgePts)){se[i] <- HedgeExperiment(testHedgePts[i], numberOfPaths)$sd}

plot(testHedgePts, se, xlab = "Number of Hedge Points", ylim=c(0,3),
     type="l", ylab = "Standard deviation", main = "Convergence of Hedge Error")
```

Discrete hedging of call option in the Heston model using risk minimizing delta

```
source("BlackScholesFormula.R")
source("Fourier.R")
```

```
# Parameters
r <- 0.04
theta <- 0.22^2
kappa <- 4.75
sigma <- 0.1
rho <- -0.5
S0 <- 100
V0 <- theta
T <- 1
K <- S0*exp(r*T)

nHedge <- 252*T
nPath <- 1000
St <-rep(S0, length = nPath)
Vt <- rep(V0, length(nPath))
dt <- T/nHedge

# Hedge
initialoutlay <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = sigma,rho,greek=1)
pfValue <-rep(initialoutlay, length = nPath)

delta <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = sigma,rho,greek=2)
vega <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = sigma,rho,greek=4)
a <- delta + (sigma*rho*vega)/(S0)
b <- pfValue - a*St

set.seed(24)
for (i in 2:nHedge){
  dW1<-sqrt(dt)*rnorm(nPath,0,1)
  dW2<-rho*dW1+sqrt(1-rho^2)*sqrt(dt)*rnorm(nPath,0,1)
  St<-St*exp((r-0.5*pmax(Vt,0))*dt+sqrt(pmax(Vt,0))*dW1)
  Vt<-Vt+kappa*(theta-pmax(Vt,0))*dt+sigma*sqrt(pmax(Vt,0))*dW2
  pfValue <- a*St+b*exp(r*dt)
  delta <-mapply(FUN=Heston.Fourier, spot=St, V =Vt, MoreArgs = list(timetoexp=T-(i-1)*dt,strike=K,
  r=r,divyield=0,theta=theta,kappa=kappa,epsilon = sigma,rho=rho,greek=2))
  vega <- mapply(FUN=Heston.Fourier, spot=St, V =Vt, MoreArgs = list(timetoexp=T-(i-1)*dt,strike=K,
```

```
  r=r,divyield=0,theta=theta,kappa=kappa,epsilon = sigma,rho=rho,greek=4))
  a <- delta + (sigma*rho*vega)/(St)
  b <- pfValue - a*St
}

plot(St, pfValue, col="blue", xlab ="S(T)", ylab =" Value of hedge portfolio ", ylim=c(-10, 100),
xlim=c(50, 150), main = "Discrete hedging of call option using Minimizing-Delta")
points(50:200, (50:200 - K)*(50:200 >= K), type ='l', lwd =5)


## Hedge Experiment
HedgeExperiment <- function(nHedge, nPath){

  St <-rep(S0, length = nPath)
  Vt <- rep(V0, length(nPath))
  dt <- T/nHedge

  initialoutlay <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = r,rho,greek=1)
  pfValue <-rep(initialoutlay, length = nPath)

  delta <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = sigma,rho,greek=2)
  vega <- Heston.Fourier(S0,T,K,r,0,V0,theta,kappa,epsilon = sigma,rho,greek=4)
  a <- delta + (sigma*rho*vega)/(S0)
  b <- pfValue - a*St

  for (i in 2:nHedge){
    dW1<-sqrt(dt)*rnorm(nPath,0,1)
    dW2<-rho*dW1+sqrt(1-rho^2)*sqrt(dt)*rnorm(nPath,0,1)
    St<-St*exp((r-0.5*pmax(Vt,0))*dt+sqrt(pmax(Vt,0))*dW1)
    Vt<-Vt+kappa*(theta-pmax(Vt,0))*dt+sigma*sqrt(pmax(Vt,0))*dW2
    pfValue <- a*St+b*exp(r*dt)
    delta <-mapply(FUN=Heston.Fourier, spot=St, V =Vt, MoreArgs = list(timetoexp=T-(i-1)*dt,strike=K
    ,r=r,divyield=0,theta=theta,kappa=kappa,epsilon = sigma,rho=rho,greek=2))
    vega <- mapply(FUN=Heston.Fourier, spot=St, V =Vt, MoreArgs = list(timetoexp=T-(i-1)*dt,strike=K
    ,r=r,divyield=0,theta=theta,kappa=kappa,epsilon = sigma,rho=rho,greek=4))
    a <- delta + (sigma*rho*vega)/(St)
```

```
    b <- pfValue - a*St
  }

  Payoff <- (St - K)*(St >= K)
  hedgeError <- pfValue - Payoff

  output = list( sd = sd(hedgeError), mean = mean(hedgeError) )
}

# table
(HedgeExperiment_table <- data.frame(c(252, 500, 1000,2000),
                                     rbind(HedgeExperiment(252*T, 100),
                                           HedgeExperiment(500*T, 100),
                                           HedgeExperiment(1000*T, 100),
                                           HedgeExperiment(2000*T, 100))))
# plot
numberOfPaths <- 10
testHedgePts <- NA
for (j in 1:1000){testHedgePts[j] <- j}
se <- numeric(length(testHedgePts))

for (i in 1:length(testHedgePts)){se[i] <- HedgeExperiment(testHedgePts[i], numberOfPaths)$sd}

plot(testHedgePts, se, xlab = "Number of Hedge Points", ylim=c(0,3),
     type="l", ylab = "Standard deviation", main = "Convergence of Hedge Error")
```