Pricing Arithmetic Asian Options using Moment Matching

Posted on **December 14, 2017** by **WilsonMongwe** —

Asian options are path-dependent options whose payoff depends on the average value of the underlying asset during a specific set of dates across the life of the option. Because the payoff of the Asian options depends on the average value of the underlying asset, volatility in the average value is lower than that of the plain vanilla options. Thus Asian options tend are less expensive than the comparable plain vanilla options. This makes Asian options ideal for use in hedging positions.

The averaging in Asian options can either be arithmetic or geometric. Under the Black Scholes world, Closed form expressions exist for geometric Asian options, as the product of log normal random variables is a log normal random variable, while approximations are required to price arithmetic Asian options as the sum of log normal random variables is not log normal.

In this blog post we will be pricing continuously sampled arithmetic Asian options using moment matching under the Black Scholes framework. We compare this moment matching with a Monte Carlo simulation.

The Asian Option

There are two basic forms of averages in Asian options, being arithmetic and geometric. A geometric average Asian option is easy to price because a closed-form solution is available. However the most difficult one to price is the arithmetic type, which is the most commonly used> No analytic solution exist for arithmetic Asian options. This missing solution is primarily because the arithmetic average of a set of log-normal random variables is not log-normally distributed.

Since no general analytical solution for the price of the arithmetic Asian option is known, a variety of techniques have been developed to approach the problem of its valuation. These include using partial differential equations, Monte Carlo simulation and moment matching among others. In this blog post we will be focusing on the last two approaches.

The Arithmetic Asian Option Payoff

As discussed in a previous blog post, diffusion models are continuous-time stochastic processes. In this blog post, we will be focusing on the GBM model. The SDE, under the risk-neutral measure, is given as

$$dS_t = S_t(rdt + \sigma dW_t) \dots (1)$$

The payoff of the fixed strike Asian option at maturity T
given as

$$\max(A_T - K, 0)$$

where $A_T = \frac{\int_0^T S_t dt}{T}$
and K
is the strike price. The value of a call option V_0
at time zero is then given as

$$V_0 = \exp(-rT) \times E^Q[\max(A_T - K, 0)]$$

where r
is the deterministic risk free rate.

If the distribution of A_T
was known, then calculating V_0
would be straight forward. However, given that we do not know the exact distribution of A_T
, we approximate it with another distribution by matching the first couple of moments of A_T
with the corresponding moments from another distribution.

In the next section we match the first two moments of A_T
with the fist two moments of the log-normal distribution.

Moment Matching

The first two raw moments of A_T
are given as :

$$m_1 = E[A_T] = \frac{S_0(\exp(rT) - 1)}{rT}$$

$$m_2 = E[A_T^2] = \frac{2S_0^2}{T} \times \left(\frac{\exp((2r + \sigma^2)T)}{(r + \sigma^2)(2r + \sigma^2)} + \frac{1}{r} \left(\frac{1}{(2r + \sigma^2)} - \frac{\exp(rT)}{r + \sigma^2} \right) \right)$$

$$\text{for } r > 0$$

The derivation for the above two equations can be found in this paper.

Recall that if Z
is $Normal(\mu, \sigma^2)$
, then $X = exp(Z)$
will be log-normally distributed. The first two raw moments of X
are then given as:

$$E[X] = \exp(\mu + 0.5\sigma^2)$$

$$E[X^2] = \exp(2\mu + 2\sigma^2).$$

Setting $E[X] = m_1$
and $E[X] = m_2$
and then solving the two equations simultaneously for μ
and σ
we have that:

$$\mu = \ln\left(\frac{m_1^2}{\sqrt{m_2}}\right)$$

$$\sigma = \sqrt{\ln\left(\frac{m_2}{m_1^2}\right)}$$

This means that:

$$V_0 = \exp(-rT) \times E^Q[\max(A_T - K, 0)]$$

$$V_0 \approx \exp(-rT) \times E^Q[\max(\exp(X) - K, 0)]$$

$$V_0 = \exp(\mu - rT + 0.5\sigma^2)\Phi(\frac{\mu - \ln K + \sigma^2}{\sigma}) - K\exp(-rT)\Phi(\frac{\mu - \ln K}{\sigma})$$

where μ
and σ
are the parameters of the approximate log-normal distribution.

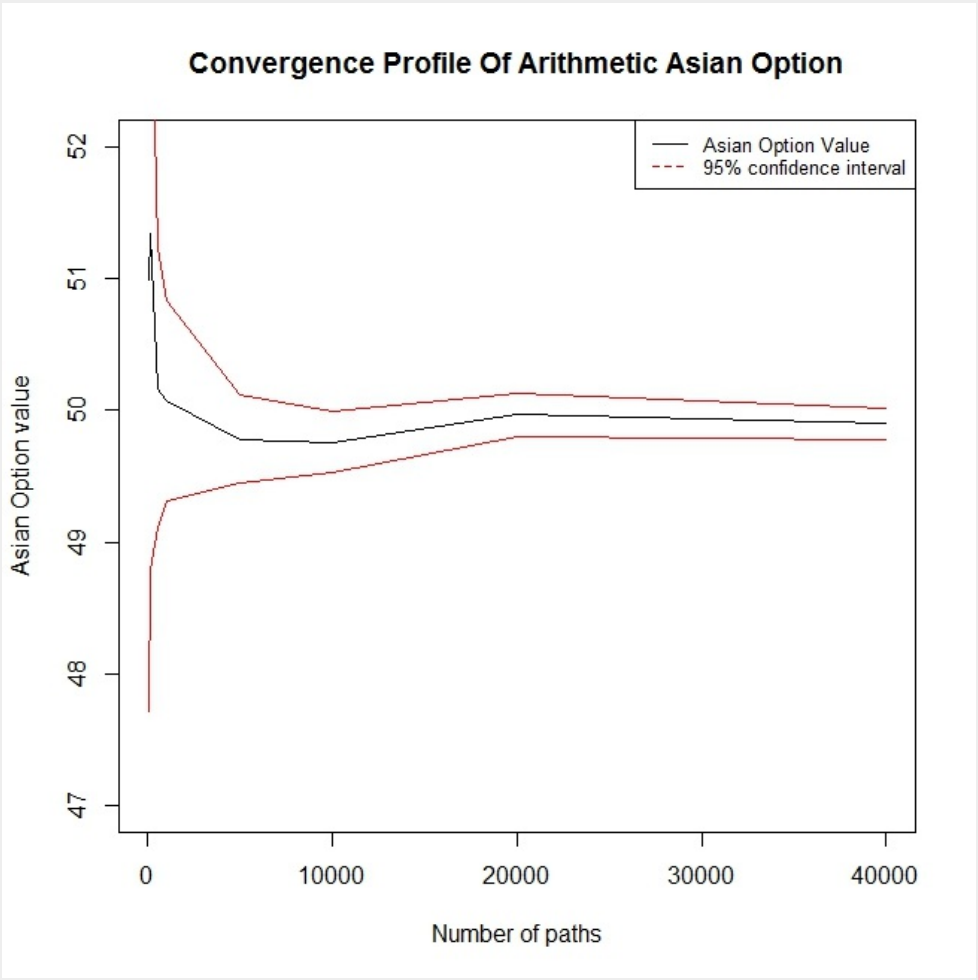
In the following section we will asses the accuracy of this approximation relative to a Monte Carlo estimate.

Simulation Study

In this section we provide the results that compare the Monte Carlo estimate and the moment matching estimate of the arithmetic Asian option described above. The Monte Carlo simulation was performed with the antithetic variates variance reduction technique.

The simulation was performed with the following parameters: $r = 0.09$
, $\sigma = 0.3$
, $T = 1$
, $S_0 = 100$
and $dt = 1/10000$
. The simulation was performed with a different number of simulations and strike prices.

The convergence profile for the Monte Carlo estimate for the price of an arithmetic Asian option with strike 50 is shown below:



— Convergence profile for the Monte Carlo price of an arithmetic Asian Option with strike at 50.

The above graph shows that the estimate becomes stable, with narrow confidence bounds, beyond 30 000 simulations.

A comparison between the Monte Carlo estimate, with 40 000 simulations, and the log-normal approximation for the arithmetic Asian Option is shown below:

Strike	Monte Carlo Estimate	Log Normal estimate	% Difference	95% Monte Carlo Confidence bounds
0	95.59352	95.63202	0.0403%	[95.48, 95.71]
50	49.89697	49.93549	0.0772%	[49.78, 50.02]
90	14.97613	15.06704	0.6069%	[14.87, 15.08]
100	8.831941	8.885762	0.6094%	[8.75, 8.92]
110	4.700716	4.69511	0.1192%	[4.63, 4.77]
150	0.1835182	0.149526	18.522%	[0.17, 0.20]
200	0.0021867	0.000639	70.745%	[0.000874, 0.0035]

The above table suggests that the approximation is good for at the money and deep in the money options, but is a very bad approximation for deep out of the money options. A better approximation, by matching more than two moments of A_T , is needed for the deep out of the money options. A shifted log normal distribution may provide better relates for the deep out of the money options.

Code Used

The code given below was used in this blog post.

```r
first_moment<-function(s_0,r,TotalTime)
{
  (s_0*(exp(r*TotalTime)-1))/(r*TotalTime)
}

second_moment<-function(s_0,r,s,TotalTime)
{
  part1 = (2*s_0^2)/(r*(r+s^2)*(2*r+s^2)*TotalTime^2)

  part2 = r*exp((2*r+s^2)*TotalTime)-(2*r+s^2)*exp(r*TotalTime)+r+s^2

  part1*part2
}
```

moments.R hosted with ❤ by **GitHub** **view raw**

```r
simulateGBM=function(s_0,drift,s,Time,dt)
{
  time_points= seq(from=0,to=Time,by=dt)
  stock=seq(from=0,to=0,length=length(time_points))
  stock_two=seq(from=0,to=0,length=length(time_points))

  #for antithetic variables
  stock[1]= log(s_0)
  stock_two[1]= log(s_0)

  for(i in 2:length(time_points))
  {
    z=rnorm(1)
    stock[i]=stock[i-1]+(drift-0.5*s^2)*dt+s*sqrt(dt)*z
    stock_two[i]=stock_two[i-1]+(drift-0.5*s^2)*dt+s*sqrt(dt)*(rnorm(1))*(-z)
  }

  return(cbind(exp(stock),exp(stock_two)))
}

```

simulateGBM.R hosted with ❤ by **GitHub** **view raw**

```r
# Parameters
s_0 = 100
s = 0.3
TotalTime = 1
delta = 0.0001
r = 0.09
time_points = seq(from=0,to=TotalTime,by=delta)

## Set up for the monte carlo simulation
total_paths = c(50,100,500,1000,5000,10000,20000,40000,100000)
strike_vector = c(0,50,90,100,110,150,200)
result = matrix(data = 0, nrow = length(total_paths), ncol = length(strike_vector))
std_dev = matrix(data = 0, nrow = length(total_paths), ncol = length(strike_vector))

lowr_bnd = matrix(data = 0, nrow = length(total_paths), ncol = length(strike_vector))
uppr_bnd = matrix(data = 0, nrow = length(total_paths), ncol = length(strike_vector))

for(path_number in 1:length(total_paths))
{
  no_paths= total_paths[path_number]
  paths= matrix(data = 0, nrow = length(time_points), ncol = no_paths)
```

```r
  paths_two= matrix(data = 0, nrow = length(time_points), ncol = no_paths)

  for(i in 1:no_paths)
  {
    paths[,i]=simulateGBM(s_0,r,s,TotalTime,delta)[,1]
    paths_two[,i]=simulateGBM(s_0,r,s,TotalTime,delta)[,2]

  }

  for(m in 1:length(strike_vector))
  {
    Strike = strike_vector[m]
    one = mean(pmax(colMeans(paths)-Strike,0))*exp(-r*TotalTime)
    two = mean(pmax(colMeans(paths_two)-Strike,0))*exp(-r*TotalTime)
    result[path_number,m] = (one+two)/2

    sum = (pmax(colMeans(paths)-Strike,0)+pmax(colMeans(paths_two)-Strike,0)) *exp(-r*TotalTime)/2
    std_dev[path_number,m] = sd(sum)

    lowr_bnd[path_number,m]= result[path_number,m]-1.96*std_dev[path_number,m] /sqrt(no_paths)
    uppr_bnd[path_number,m]= result[path_number,m]+1.96*std_dev[path_number,m] /sqrt(no_paths)
  }

  print(paste("Path Index  :", path_number, " of ", length(total_paths)))
  print(paste("Number of paths :", no_paths))
}


### Example convergence profile of Monte Carlo
x_m = result[1:8,2]
lower = lowr_bnd[1:8,2]
upper = uppr_bnd[1:8,2]

xvals= total_paths[1:8]#seq(from=100,to=100*total_paths,by=100)
plot(xvals,x_m,type="l",ylim=c(47,52),xlab="Number of paths",ylab= "Asian Option value",
     main="Convergence Profile Of Arithmetic Asian Option")
lines(xvals,lower,type="l",col="red")
lines(xvals,upper,type="l",col="red")
legend("topright", legend=c("Asian Option Value","95% confidence interval"),
       col=c("black","red"), lty=1:2, cex=0.8)

###### Log normal approximation
m1 = first_moment(s_0,r,TotalTime)
m2 = second_moment(s_0,r,s,TotalTime)

mu_of_average = log(m1^2/sqrt(m2))
sig_of_average = sqrt(log(m2/m1^2))

ln_result= seq(from=0,to=0,length=length(strike_vector))

for(i in 1:length(ln_result))
{
  ln_result[i]=price_option(mu_of_average,r,TotalTime,sig_of_average,strike_vector[i])
}

### Percentage difference between the best Monte Carlo and Log Normal approximation
((result[8,]-ln_result)/result[8,])*100
```

Related

An Interactive Dynamic Delta Hedging Example in R
August 27, 2017
In "Computer Science"

Introduction to Diffusion and Jump Diffusion Processes
March 4, 2015
In "Mathematical Finance"

Calibrating Financial Models using a Non-Parametric Technique
September 30, 2017
In "Mathematical Finance"

This entry was posted in **Mathematical Finance**, **Statistics** and tagged **Diffusion**, **Hedging**, **Option**, **Statistics** by **WilsonMongwe**. Bookmark the **permalink [http://www.wilsonmongwe.co.za/pricing-arithmetic-asian-options-using-moment-matching/]** .