Asian Option Pricing: Monte Carlo Control Variate

A discrete arithmetic Asian call option has the payoff

$$\left(\frac{1}{N+1} \sum_{i=0}^{N} S_{\frac{T_i}{N}} - K \right)^+$$

A discrete geometric Asian call option has the payoff

$$\left(\left[\prod_{i=0}^{N} S_{\frac{T_i}{N}} \right]^{\frac{1}{N+1}} - K \right)^+$$

It is known that in the Black-Scholes model the price of the geometric Asian call option is given by

$$e^{-rT}(S_0 e^{\rho T} N(d_1) - K N(d_2))$$

where

$$\rho := \frac{1}{2}\left(r - \frac{1}{2}\sigma^2 + \hat{\sigma}^2 \right)$$

$$\hat{\sigma} := \sigma \sqrt{\frac{2N+1}{6(N+1)}}$$

$$d_1 := \frac{1}{\sqrt{T}\hat{\sigma}} \left(\ln(S_0/K) + \left(\rho + \frac{1}{2}\hat{\sigma}^2 \right) T \right)$$

$$d_2 := \frac{1}{\sqrt{T}\hat{\sigma}} \left(\ln(S_0/K) + \left(\rho - \frac{1}{2}\hat{\sigma}^2 \right) T \right)$$

The price of an arithmetic Asian option can be computed by Monte Carlo simulation. However, in order to achieve a small standard error, the number of simulations must be very high - somewhere in the neighborhood of 100,000. This can be very computationally expensive. However, if we use the payoff of the geometric Asian option as the control variate, this will allow us to obtain the same level of accuracy with fewer simulations and therefore less time. Here is the MATLAB implementation:

```
function price = geoAsianOpt(S0,sigma,K,r,T,Nt,type)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This is a closed form solution for geometric Asian options
%
%S0     = Current price of underlying asset
%sigma  = Volatility
%K      = Strike price
%r      = Risk-free rate
%T      = Time to maturity
%Nt     = Time intervals
%type   (0 for call, 1 for put)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

adj_sigma=sigma*sqrt((2*Nt+1)/(6*(Nt+1)));
rho=0.5*(r-(sigma^2)*0.5+adj_sigma^2);

d1 = (log(S0/K)+(rho+0.5*adj_sigma^2)*T)/(adj_sigma*sqrt(T));
d2 = (log(S0/K)+(rho-0.5*adj_sigma^2)*T)/(adj_sigma*sqrt(T));

if type==0
  price = exp(-r*T)*(S0*exp(rho*T)*normcdf(d1)-K*normcdf(d2));
else
  price = exp(-r*T)*(K*normcdf(-d2)-S0*exp(rho*T)*normcdf(-d1));
end
```

```
function [price error] = MCCVAsianOpt(S0,sigma,K,r,T,Nt,Nc,Nruns,type)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This is a Monte Carlo pricing algorithm using control variates
%for arithmetic Asian options. The control variate used is the price of
%a geometric Asian option.
%
%S0     = Current price of underlying asset
%sigma  = Volatility
%K      = Strike price
%r      = Risk-free rate
%T      = Time to maturity
%Nt     = Time intervals
%Nruns  = Number of simulations
%type   (0 for call, 1 for put)
%The payoff function of this option is represented by:
%        Max{0, (Average price of underlying asset - Strike price)}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%_____________________________________________________________________________
%                           Random Number Generator
%_____________________________________________________________________________
%Generates random numbers from the standard normal distribution
dB = randn(Nt,Nruns);

%_____________________________________________________________________________
%                  Underlying Asset Price Process Simulation
%_____________________________________________________________________________
%time
%dt=1/250;
%T=days*dt;
dt=T/Nt;

%Calculates the expected return
k = r - (sigma^2)*0.5;

%Calculates the deterministic component of the price process
deterministic = repmat(k * dt * (1:Nt)',1,Nruns);

%Calculates the stochastic component of the price process
stochastic = sigma*sqrt(dt).*cumsum(dB);

paths = [repmat(S0,1,Nruns); S0 * exp(deterministic + stochastic)];


%_____________________________________________________________________________
%                            Control Variates
%_____________________________________________________________________________

divisor=1/(Nt+1);
DF = exp(-r*T);

%price of geometric mean Asian option using formula
geoExact = geoAsianOpt(S0,sigma,K,r,T,Nt,type);

geoCallPrices = zeros(Nc,1);
geoPutPrices = zeros(Nc,1);
ariCallPrices = zeros(Nc,1);
ariPutPrices = zeros(Nc,1);

%repeated runs to find control parameter
for i=1:Nc;
    pathVector = paths(:,i); %creates vector for each price path
    avgPathPrice = sum(pathVector)*divisor;
    if type == 0
```

```
       geoCallPrices(i) = DF*max(geomean(pathVector) - K,0);
       ariCallPrices(i) = DF*max(avgPathPrice - K,0);
    else
       geoPutPrices(i) = DF*max(K - geomean(pathVector),0);
       ariPutPrices(i) = DF*max(K - avgPathPrice,0);
    end
end

%finding optimal c
if type == 0
     %variance covariance matrix
     MatCov = cov(geoCallPrices, ariCallPrices);
     %c=Cov(geoCallPrices,ariCallPrices)/Var(geoCallPrices)
     c = -MatCov(1,2)/var(geoCallPrices);
else
     %variance covariance matrix
     MatCov = cov(geoPutPrices, ariPutPrices);
     %c=Cov(geoPutPrices,ariPutPrices)/Var(geoPutPrices)
     c = -MatCov(1,2)/var(geoPutPrices);
end

controlVars=zeros(Nruns,1);

%_____________________________________________________________________________
%                         Monte Carlo Algorithm
%_____________________________________________________________________________

for i=1:Nruns;
    pathVector = paths(:,i); %creates vector for each price path
    avgPathPrice = sum(pathVector)*divisor;
    if type == 0
       geoCallPrice = DF*max(geomean(pathVector) - K,0);
       ariCallPrice = DF*max(avgPathPrice - K,0);
       %adjusted payoff using control variate
       controlVars(i) = ariCallPrice + c * (geoCallPrice - geoExact);
    else
       geoPutPrice = DF*max(K - geomean(pathVector),0);
       ariPutPrice = DF*max(K - avgPathPrice,0);
       %adjusted payoff - using control variate
       controlVars(i) = ariPutPrice + c * (geoPutPrice - geoExact);
    end
end



price = mean(controlVars);
```

```
error = std(controlVars)/sqrt(Nruns);



%program that runs the above functions
clear;

S0=100;
V=0.3;
K=100;
r=0.03;
T=5;
Nt=T*252; %number of trading days until maturity
Nc=1000;
Nruns=100000; %monte carlo runs for standard method
NrunsCV=10000; %monte carlo runs for control variate method
type=1; %0 for calls; 1 for puts

%Arithmetic Asian option price using Monte Carlo
%[arithmeticAsianPrice error] = MC_AsianOpt(S0,V,K,r,T,Nt,Nruns,type)

%Geometric Asian option price using formula
%geoAsianExactPrice = geoAsianOpt(S0,V,K,r,T,Nt,type)

%Geometric Asian option price using Monte Carlo
%[geoAsianPriceMC error] = geoMCAsianOpt(S0,V,K,r,T,Nt,Nruns,type)

%Arithmetic Asian option price using Monte Carlo Control Variate
[ariAsianPriceMCCV error] = MCCVAsianOpt(S0,V,K,r,T,Nt,Nc,NrunsCV,type)

%Efficiency Comparison:
%in this example, to get a similar error without control variates,
%you need about 800 more simulations than you do with control variate.
```