Department of Economics, Mathematics and Statistics

MSc in Financial Engineering

On Numerical Methods for the Pricing of Commodity Spread Options

Damien Deville

September 11, 2009

Supervisor: Dr. Steve Ohana

"This report is submitted as part requirement for the MSc Degree in Financial Engineering at Birkbeck College. It is substantially the result of my own work except where explicitly indicated in the text."

Abstract

We study numerical methods to price commodity spread options. We first present spread options and particularly the crack spread options paying the buyer the difference between the crude oil and heating oil futures prices. We then present the Monte Carlo option pricing pricing framework and discuss its application to spread option pricing. Finally, we discuss some variance reduction techniques and show their efficacy in front of a classic Monte Carlo computation.

A copy of this report and all the code written for the project can be found on my website:

http://www.ddeville.me

Contents

1 An overview of the Spread options **4**

1.1 The spread options in commodities 4

1.2 The case of the Crack Spread option 5

1.3 The Kirk formula 6

2 Monte Carlo framework for pricing options **8**

2.1 Results from probability theory 9

2.2 The discretization method: Euler scheme 10

2.3 The spread option setting 12

3 Variance reduction techniques **15**

3.1 Antithetic variates 15

3.2 Control variates 18

4 Results **23**

5 Conclusion **25**

Appendix **26**

A Additional Matlab code **26**

References **28**

1 An overview of the Spread options

A spread option is an option that pays the buyer at maturity the difference between two assets, generally futures prices. However, spread options can be written on all types of financial products including equities, bonds and currencies.
If we assume a spread option between two assets S_1 and S_2, at maturity the payoffs or both call and put will be

$$f_C = \max(S_1 - S_2 - K, 0) \tag{1}$$

$$f_P = \max(K - S_1 + S_2, 0) \tag{2}$$

where

- f_C is the price of the call spread option at maturity
- f_P is the price of the put spread option at maturity
- S_1 is the price of asset 1 at maturity
- S_2 is the price of asset 2 at maturity
- K is the strike price

1.1 The spread options in commodities

The spread options are among the most traded instruments in the world of commodities [9]. Generally, the payoff of a spread option is defined as the difference between two or more commodities.
However, it can also be defined as the difference between two different maturities for the same commodity. In this case, it is called a calendar spread. Calendar spread are used in order to hedge risks linked to the seasonality of a commodity.
In this project, we will focus on the spreads between two commodities and particularly the energy commodities.
Some types of commodity spreads enable the trader to gain exposure to the commodity's

production process. This is created by purchasing a spread option based on the difference between the inputs and outputs of the process. Common examples of this type of spread are the crack, crush and spark spreads.

1.2 The case of the Crack Spread option

A petroleum refiner is caught between two markets: the crude oil market from which he buys raw materials and whose prices determine a large amount of its costs and the refined product (heating oil or gasoline) market where he sells its production and whose prices determine its earning. A change in the price difference between these two prices can represent an enormous risk for the refiner. There has been a tremendous demand for financial products able to cope with this problem and offer way to hedge this risk for refiners.
Crack Spread options on oil products are traded on the New York Mercantile Exchange. Crack spread options are puts and calls on the one-to-one ratio between the New York Harbor heating oil futures and New York Harbor unleaded gasoline futures contract and the Exchange's light, sweet crude oil futures contract. The underlying spread and options price is expressed as dollars and cents per barrel. The one-to-one ratio of the options meets the needs of many refiners, because it reflects the refiner's exposure related to the manufacture of gasoline and heating oil throughout the year.

The most popular crack spread is the *3-2-1 crack spread*. Its payoff at maturity is defined by

$$f_{CS} = 3f_{CO} - 2f_{GA} - 1f_{HO} \tag{3}$$

where

- f_{CS} is the crack spread value at maturity
- f_{CO} is the value of a crude oil future contract at maturity
- f_{GA} is the value of a gasoline future contract at maturity
- f_{HO} is the value of a heating oil future contract at maturity

However, even if the *3-2-1 crack spread* is the most famous among the future contracts, when dealing with spread options, we usually consider spreads between only two underlying future prices.

1.3 The Kirk formula

In 1995, Kirk gave an approximation and proposed a pricing formula for a spread options between two underlying futures contracts assuming both underlying assets follow correlated geometric Brownian motion. The formula for both call and put spread options are as follows

$$f^C(0) = (S_2 + K)\left[e^{-rT}\left(\Phi(d_1)\frac{S_1}{S_2+K} - \Phi(d_2)\right)\right] \tag{4}$$

$$f^P(0) = (S_2 + K)\left[e^{-rT}\left(\Phi(-d_2) - \Phi(-d_1)\frac{S_1}{S_2+K}\right)\right] \tag{5}$$

where

$$d_1 = \frac{\log\left(\frac{S_1}{S_2+K}\right) + (\sigma^2/2)T}{\sigma\sqrt{T}} \tag{6}$$

$$d_2 = \frac{\log\left(\frac{S_1}{S_2+K}\right) - (\sigma^2/2)T}{\sigma\sqrt{T}} \tag{7}$$

and

$$\sigma = \sqrt{\sigma_1^2 + \left(\sigma_2\frac{S_2}{S_2+K}\right)^2 - 2\rho\sigma_1\sigma_2\frac{S_2}{S_2+K}} \tag{8}$$

where

- $f(0)$ is the option price at date 0
- r is the risk-free interest rate
- $\Phi(T)$ is the cumulative standard Normal
- $S_{1,2}$ is the asset price 1,2
- K is the strike price
- $\sigma_{1,2}$ is the volatility of asset 1,2

- ρ is the correlation

We will use this approximation to check the goodness of the Monte Carlo methods we will use.

In Listing 1, we show the implementation of the Kirk approximation in Matlab.

```
function [call,put] = kirk(K, S1, S2, sigma1, sigma2, rho, r, T)
    % K = strike price
    % S1 = spot price for asset 1
    % S2 = spot price for asset 2
    % sigma1 = volatility for asset 1
    % sigma2 = volatility for asset 2
    % rho = correlation between assets
    % r = risk-free rate
    % T = time to maturity

    frac1 = S1/(S2+K) ;
    frac2 = S2/(S2+K) ;
    sigma = sqrt(sigma1^2 + sigma2^2*frac2^2 - 2*rho*sigma1*sigma2*frac2) ;

    d1 = (log(frac1) + 0.5*(sigma^2)*T)/(sigma*sqrt(T)) ;
    d2 = (log(frac1) - 0.5*(sigma^2)*T)/(sigma*sqrt(T)) ;

    call = (S2+K)*(exp(-r*T)*(frac1*normcdf(d1) - normcdf(d2))) ;
    put = (S2+K)*(exp(-r*T)*(normcdf(-d2) - frac1*normcdf(-d1))) ;

end
```

Listing 1: The implementation of the Kirk formula in Matlab

2 Monte Carlo framework for pricing options

When pricing exotic options, very often a closed-form solution is not available. We thus have to use some numerical methods. In the case of spread options, the following numerical methods are available:

- Binomial or trinomial trees
- Finite differences (PDE solver)
- Monte Carlo methods
- FFT methods (in case we know the characteristic function for the underlying process, which is always true with Lévy processes for example)

For the scope of this project, we will focus on Monte Carlo methods.
Monte Carlo methods are famous for their simplicity but unfortunately also for their slowness. However, a large number of variance reduction techniques exist that allow the method to converge faster and then require less steps to be used.
Following, we present the basics of the Monte Carlo method.

When pricing an option under the Monte Carlo framework, we use the fact that an option price can be written as the following discounted expectation

$$f(t) = e^{-r(T-t)}\mathbb{E}^{\mathbb{Q}}\left[\phi(T)|\mathcal{F}_t\right] \tag{9}$$

where

- $f(t)$ is the option price at date $t < T$
- r is the risk-free interest rate
- $\phi(T)$ is the option payoff at date T
- $\mathcal{F}_t$ is the filtration representing the information set available at date t
- $\mathbb{E}^{\mathbb{Q}}$ is the expectation under the risk-neutral measure $\mathbb{Q}$

2.1 Results from probability theory

By the *law of large numbers*, we know that the sample average converges, in the limit, to the expected value

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n} X_i = \mathbb{E}(X) \tag{10}$$

Thus we can write

$$e^{-r(T-t)}\mathbb{E}^{\mathbb{Q}}\left(\phi(T)|\mathcal{F}_t\right) = e^{-r(T-t)} \lim_{n\to\infty} \left(\frac{1}{n} \sum_{i=0}^{n} \phi_i(T) \right) \tag{11}$$

Thus, if we insure that n is big enough, we can approximate the expectation quite accurately by the sample mean. The Monte Carlo method is thus based on simulating a large number of sample price paths used to compute the payoffs. Then, by taking the sample mean among these payoffs, we can approximate the expectation in Equation 9. Finally, by discounting, we get the option price.

Similarly, by the *central limit theorem*, we know that the sum of a sufficiently large number of independently generated random numbers X_i will be approximately Normally distributed. Assuming that σ is the standard deviation of these independently generated random numbers and $\bar{X}$ is the mean. We also define $\epsilon = \bar{X} - \mathbb{E}(X)$ as the estimation error. Then, $\frac{\epsilon\sqrt{n}}{\sigma}$ converges toward a standard Normal distribution. We also know that if $Z \sim \mathbb{N}(0,1)$, we have $\mathbb{P}\left(|Z| \leq 1.96\right) = \alpha$ where $\alpha = 0.95$. Therefore

$$\mathbb{P}\left(\frac{\epsilon\sqrt{n}}{\sigma} \leq 1.96 \right) = 0.95 \tag{12}$$

$$\mathbb{P}\left(\epsilon \leq 1.96\frac{\sigma}{\sqrt{n}} \right) = 0.95 \tag{13}$$

$$\mathbb{P}\left(\bar{X} - \mathbb{E}(X) \leq 1.96\frac{\sigma}{\sqrt{n}} \right) = 0.95 \tag{14}$$

Then, a 95% confidence interval for $\mathbb{E}(X)$ is given by

$$\left(\bar{X} - 1.96\frac{\sigma}{\sqrt{n}}; \bar{X} + 1.96\frac{\sigma}{\sqrt{n}} \right) \tag{15}$$

but we also have to bear in mind that we do not know σ, so we need to use the estimator

$$\sigma_n = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2} \tag{16}$$

and the 95% confidence interval for $\mathbb{E}(X)$ is thus given by

$$\left(\bar{X} - 1.96\frac{\sigma_n}{\sqrt{n}}; \bar{X} + 1.96\frac{\sigma_n}{\sqrt{n}}\right) \tag{17}$$

2.2 The discretization method: Euler scheme

In order to simulate the underlying stochastic differential equation, we need to discretize it. A result from this discretization is the discretization-error which is added to the Monte-Carlo error. One of the most used discretization scheme in financial engineering is the Euler scheme.

Consider the following stochastic differential equation

$$dX(t) = a(X,t)dt + b(X,t)dW(t) \tag{18}$$

then the Euler scheme states that, assuming a time grid $0 = t_0 < t_1 < \ldots < t_{n-1} < t_n = t$, the following discretization can be applied

$$X(t_{i+1}) - X(t_i) = a(X,t_i)(t_{i+1} - t_i) + b(X,t_i)\sqrt{t_{i+1} - t_i}Z_{i+1} \tag{19}$$

or

$$X(t_{i+1}) = X(t_i) + a(X,t_i)(t_{i+1} - t_i) + b(X,t_i)\sqrt{t_{i+1} - t_i}Z_{i+1} \tag{20}$$

where Z_i is a standard normal random variable. Assuming a fixed grid spacing $t_{i+1} - t_i = h$, we thus have $t_i = ih$ and we can rewrite

$$X(i+1) = X(i) + a(X(i))h + b(X(i))\sqrt{h}Z_{i+1} \tag{21}$$

In Listing 2, we show the implementation in Matlab of the Monte Carlo method for a spread option.

```
function [call, put, conf] = BS_MC(S,K,r,sigma,T,N)
    % K = strike price
    % S = asset spot price
    % sigma = volatility
    % r = risk-free interest rate
    % T = time to maturity
    % N = number of simulations

    % We generate a standard Gaussian vectors
    Z = randn(N,1) ;

    % We generate final asset prices from the random vector
    S_fin = S * exp( (r-0.5*sigma^2)*T + sigma*sqrt(T)*Z ) ;

    % We compute the payoff vector for both call and put
    resC = max(S_fin - K, 0) ;
    resP = max(K - S_fin, 0) ;

    % We finally discount the average of the payoff
    call = exp(-r*T) * mean(resC) ;
    put = exp(-r*T) * mean(resP) ;

    % We compute the variance and the confidence intervall
    var = sum((resC-mean(resC).^2)/(N-1) ;
    conf = 1.96*var/sqrt(N) ;
end
```

Listing 2: Implementation of the classic Monte Carlo routine for pricing a vanilla option in Matlab

2.3 The spread option setting

When pricing spread options with the Monte Carlo method, we used the fact that the dynamics of both asset prices can be written as

$$\begin{aligned} \frac{dS_1(t)}{S_1(t)} &= \mu_1(t, S_1(t), S_2(t))dt + \sigma_1(t, S_1(t), S_2(t)) \\ &\quad \Big[\rho(t, S_1(t), S_2(t))dW_1(t) + \sqrt{1-\rho(t, S_1(t), S_2(t))}dW_2(t)\Big] \end{aligned} \tag{22}$$

$$\frac{dS_2(t)}{S_2(t)} = \mu_2(t, S_1(t), S_2(t))dt + \sigma_2(t, S_1(t), S_2(t))dW_2(t) \tag{23}$$

where

- $S_{1,2}$ is the price of the asset 1, 2
- $\mu_{1,2}$ is the cost of carry for the asset 1, 2
- $\sigma_{1,2}$ is the volatility for the asset 1, 2
- ρ is the correlation between both assets

The two equations can be solved separately and the solutions are as following

$$S_1(T) = S_1(0)\exp\Big[(\mu_1 - \sigma_1^2/2)T + \sigma_1\rho\sqrt{T}U + \sigma_2\sqrt{1-\rho^2}\sqrt{T}V\Big] \tag{24}$$

$$S_2(T) = S_2(0)\exp\Big[(\mu_2 - \sigma_2^2/2)T + \sigma_2\sqrt{T}U\Big] \tag{25}$$

where U and V are two independent standard Gaussian random variables.

Then, we know that the payoff at maturity of a spread call option is defined as

$$\Phi^C(T) = S_1(T) - S_2(T) - K \tag{26}$$

and a spread put option

$$\Phi^P(T) = K - [S_1(T) - S_2(T)] \tag{27}$$

So, as explained in the previous section, given the value of the parameters and the asset prices today $S_1(0)$ and $S_2(0)$, we can compute the option price at date t by simulation a large number of asset prices $S_1(T)$ and $S_2(T)$ from Equations 24 and 25. We can then compute

the payoff from these simulations with Equations 26 and 27, take an average of it and finally discount it in order to get the option price.

In Listing 3, we show the implementation in Matlab of the Monte Carlo method for a spread option.

```
function [call, put, conf] = MC(K, S1, S2, sigma1, sigma2, mu1, mu2, rho, r, T, N)
    % K = strike price
    % S1 = spot price for asset 1
    % S2 = spot price for asset 2
    % sigma1 = volatility for asset 1
    % sigma2 = volatility for asset 2
    % mu1 = cost of carry for asset 1
    % mu2 = cost of carry for asset 2
    % rho = correlation between assets
    % r = risk-free rate
    % T = time to maturity
    % N = number of simulations

    % We generate two standard Gaussian vectors
    U = randn(N,1) ;
    V = randn(N,1) ;

    % We generate final correlated asset prices from the 2 random vectors
    S1_fin = S1 * exp( (mu1-0.5*sigma1^2)*T + sigma1*rho*sqrt(T)*U + ...
             ... + sigma2*sqrt(1-rho^2)*sqrt(T)*V ) ;
    S2_fin = S2 * exp( (mu2-0.5*sigma2^2)*T + sigma2*sqrt(T)*U ) ;

    % We compute the payoff vector for both call and put
    resC = max((S1_fin - S2_fin) - K, 0) ;
    resP = max(K - (S1_fin - S2_fin), 0) ;

    % We finally discount the average of the payoff
    call = exp(-r*T) * mean(resC) ;
    put = exp(-r*T) * mean(resP) ;

    % We compute the variance and the confidence intervall
    var = sum((resC-meanC).^2)/(N-1) ;
    conf = 1.96*var/sqrt(N) ;
end
```

Listing 3: Implementation of the classic Monte Carlo routine for pricing a spread option in Matlab

3 Variance reduction techniques

In order to increment the efficiency of Monte Carlo simulations, we need to reduce the variance of the estimates.
We are presenting two variance reduction techniques: antithetic variates and control variates. We first apply these techniques to vanilla options in order to compare their efficacy with the Black-Scholes formula.

3.1 Antithetic variates

The method of antithetic variates reduces variance by introducing negative dependence between pairs of replications.
For example, when generating random variables normally distributed, the variables Z and $-Z$ form an antithetic pair since a large value of one results in a small value of the other. This is due to the fact that the standard Normal distribution is symmetric around zero. Thus, if we use the standard $N(0,1)$ i.i.d variables Z_i in order to generate a Brownian motion path, using $-Z_i$ we simulate the reflection of this path about the origin. Based upon this fact, using this method we actually reduce the variance of the errors.

Assuming we want to price a vanilla option using the Monte Carlo numerical method coupled with the antithetic variates method in order to reduce the variance, we have to proceed as follows:

- we compute a vector $\mathbf{Z}_1$ of Normal random variables.
- we compute a vector of asset prices $\mathbf{S}_1$ under the GBM process based on the vector $\mathbf{Z}_1$
- we create a new vector $\mathbf{Z}_2$ given by $\mathbf{Z}_2 = -\mathbf{Z}_1$
- we compute a new vector of asset prices $\mathbf{S}_2$ under the GBM process based on the vector $\mathbf{Z}_2$

- we take an average of the values constituting both vectors $\mathbf{S}_1$ and $\mathbf{S}_2$ that we call A_1 and A_2
- we take the following average $(A_1 + A_2)/2$ and we discount it.

In Listing 4, we show the implementation in Matlab of the Monte Carlo method for pricing a vanilla option using the antithetic variates for reducing the variance of the estimates.

In Figure 1, we show the convergence of the Monte Carlo option prices to the actual Black-Scholes price using both the classic Monte Carlo method and the Monte Carlo method coupled with the antithetic variates. We clearly see that even if both seem to converge to the same value, the variance of the second one is lower and the price converges a lot faster.

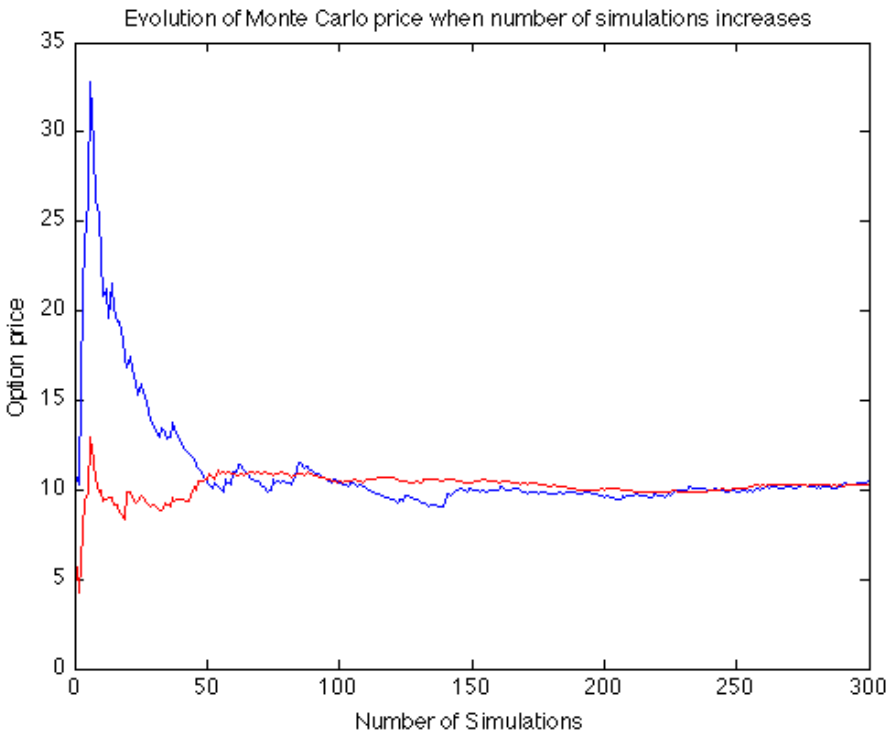

Figure 1: Evolution of the Monte Carlo option price when the number of simulations increases. Using antithetic variates (red), we see the Monte Carlo price converges earlier.

```
function [call, put, conf] = antitheticBS_MC(S,K,r,sigma,T,N)
    % S = sport price
    % K = strike price
    % r = risk-free interest rate
    % sigma = volatility
    % T = time to maturity
    % N = number of simulations

    % We generate a standard Gaussian vector
    Z = randn(N,1) ;
    % We generate the antithetic random vector
    nZ = -Z ;

    % We generate final asset prices from both random vectors
    pS_fin = S * exp((r-0.5*sigma^2)*T + sigma*sqrt(T)*Z ) ;
    nS_fin = S * exp((r-0.5*sigma^2)*T + sigma*sqrt(T)*nZ ) ;

    % We compute the payoff vector for the call for both random vectors
    resCp = max( pS_fin - K, 0) ;
    resCn = max( nS_fin - K, 0) ;
    % We compute the payoff vector for the put for both random vectors
    resPp = max(K - pS_fin , 0) ;
    resPn = max(K - nS_fin , 0) ;

    % We compute the average between normal and antithetic payoffs
    resC = 0.5 * (resCp + resCn) ;
    resP = 0.5 * (resPp + resPn) ;

    % We finally discount the average of the payoff
    call = exp(-r*T)*mean(resC) ;
    put = exp(-r*T)*mean(resP) ;

   % We compute the variance and the confidence intervall
    var = sum((resC-meanC).^2)/(N-1) ;
    conf = 1.96*var/sqrt(N) ;
end
```

Listing 4: Matlab function that computes vanilla option prices with Monte Carlo using antithetic variates variance reduction technique.

In Listing 5 we show the Matlab implementation of the Monte Carlo method for pricing a spread option using the antithetic variates variance reduction technique.

3.2 Control variates

The control variate technique allows an effective variance reduction in a simple theoretical framework. The method takes advantage of random variables with known expected value and positively correlated with the variable under consideration. Let Y be a random variable whose mean is to be determined through Monte Carlo simulation and X a random variable with known mean $\mathbb{E}(X)$. Now, for each trial, the outcome of X_i is calculated along with the output of Y_i. Let us call $\bar{Y}^{CV}$ the control variate estimator of $\mathbb{E}(Y)$. The control variate estimator is thus given by

$$\begin{aligned} \bar{Y}^{CV} &= \frac{1}{n}\sum_{i=1}^{n}(Y_i - b(X_i - \mathbb{E}(X))) && (28) \\ &= \bar{Y} - b(\bar{X} - \mathbb{E}(X)) && (29) \end{aligned}$$

for any fixed b (further we will talk about the estimation of this parameter b).
We can show that the control variate estimator $\bar{Y}_{CV}$ is unbiased and consistent

$$\mathbb{E}(\bar{Y}^{CV}) = \mathbb{E}(\bar{Y}) - \mathbb{E}(\bar{X}) + \mathbb{E}(\bar{X}) = \mathbb{E}(Y) \tag{30}$$

and

$$\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}Y_i^{CV} = \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}(Y_i - X_i + \mathbb{E}(X)) = \mathbb{E}(Y - X + \mathbb{E}(X)) \to \mathbb{E}(Y) \tag{31}$$

Then, each Y_i has variance

$$\begin{aligned} \mathrm{var}(Y_i^{CV}) &= \mathrm{var}(Y_i - b[X_i - \mathbb{E}(X)]) && (32) \\ &= \sigma_Y^2 - 2b\sigma_X\sigma_Y\rho + b^2\sigma_X^2 && (33) \end{aligned}$$

where $\sigma_X^2 = \mathrm{var}(X)$, $\sigma_Y^2 = \mathrm{var}(Y)$ and ρ is the correlation between X and Y.
The optimal coefficient b^* that minimizes the variance is given by

$$b^* = \frac{\sigma_Y}{\sigma_X}\rho = \frac{\mathrm{cov}(X,Y)}{\mathrm{var}(X)} \tag{34}$$

```
function [call, put] = antitheticMC(K, S1, S2, sigma1, sigma2, mu1, mu2, rho, r, T, N)
    % K = strike price
    % S1,2 = spot price for asset 1,2
    % sigma1,2 = volatility for asset 1,2
    % mu1,2 = cost of carry for asset 1,2
    % rho = correlation between assets
    % r = risk-free rate
    % T = time to maturity
    % N = number of simulations

    % We generate two standard Gaussian vectors
    U = randn(N,1) ; V = randn(N,1) ;
    % We generate the antithetic random vectors
    nU = -U ; nV = -V ;

    % We generate final prices for both assets from both random vectors
    pS1_fin = S1 * exp( (mu1-0.5*sigma1^2)*T + sigma1*rho*sqrt(T)*U + ...
              ... + sigma2*sqrt(1-rho^2)*sqrt(T)*V ) ;
    nS1_fin = S1 * exp( (mu1-0.5*sigma1^2)*T + sigma1*rho*sqrt(T)*nU + ...
              ... + sigma2*sqrt(1-rho^2)*sqrt(T)*nV ) ;
    pS2_fin = S2 * exp( (mu2-0.5*sigma2^2)*T + sigma2*sqrt(T)*U ) ;
    nS2_fin = S2 * exp( (mu2-0.5*sigma2^2)*T + sigma2*sqrt(T)*nU ) ;

    % We compute the payoff vector for the call for both random vectors
    resCp = max((pS1_fin - pS2_fin) - K, 0) ;
    resCn = max((nS1_fin - nS2_fin) - K, 0) ;
    % We compute the payoff vector for the put for both random vectors
    resPp = max(K - (pS1_fin - pS2_fin), 0) ;
    resPn = max(K - (nS1_fin - nS2_fin), 0) ;

    % We compute the average between normal and antithetic payoffs
    resC = 0.5 * (resCp + resCn) ;
    resP = 0.5 * (resPp + resPn) ;

    % We finally discount the average of the payoff
    call = exp(-r*T) * mean(resC) ;
    put = exp(-r*T) * mean(resP) ;
end
```

Listing 5: Matlab function that computes spread option prices with Monte Carlo using antithetic variates variance reduction technique.

Since we do not know the values of $\text{cov}(X, Y)$ and $\text{var}(X)$, we have to estimate b^* as follows

$$\widehat{b^*} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2} \tag{35}$$

As a control variate, we can use the underlying asset for example. Since we know that discounted asset prices are martingale

$$\mathbb{E}(S(T)) = e^{rT}S(0) \tag{36}$$

we can easily use S as the control variate and, assuming ϕ is the payoff of the option, we can write the control variate estimator as

$$\frac{1}{n}\sum_{i=1}^{n}(\phi_i - b[S_i(T) - e^{rT}S(0)]) \tag{37}$$

In Listing 6, we show the implementation in Matlab of the Monte Carlo method for pricing a vanilla option using the control variates for reducing the variance of the estimates.

In Figure 2, we show the convergence of the Monte Carlo option prices to the actual Black-Scholes price using both the classic Monte Carlo method and the Monte Carlo method coupled with the control variates. We clearly see that even if both seem to converge to the same value, the variance of the second one is lower and the price converges a lot faster.

```
function [call, put] = controlBS_MC(S,K,r,sigma,T,N)
    % S = sport price
    % K = strike price
    % r = risk-free interest rate
    % sigma = volatility
    % T = time to maturity
    % N = number of simulations

    % We generate a standard Gaussian vector
    Z = randn(N,1) ;

    % We generate final asset prices from the random vector
    S_fin = S*exp((r-0.5*sigma^2)*T + sigma*sqrt(T)*Z) ;

    % We compute the payoff vector for both call and put
    resC = max(S_fin - K, 0) ;
    resP = max(K - S_fin, 0) ;

    % We substract the mean to each payoff
    newC = resC - mean(resC) ;
    newP = resP - mean(resP) ;
    % We substract the mean to each final price
    newS = S_fin - mean(S_fin) ;

    % We compute the value of beta for both call and put
    bC = mean(newC.*newS)/mean(newS.*newS) ;
    bP = mean(newP.*newS)/mean(newS.*newS) ;

    % We compute the payoff control variates
    finalC = resC - bC*(S_fin - exp(r*T)*S) ;
    finalP = resP - bP*(S_fin - exp(r*T)*S) ;

    % We finally discount the average of the payoff
    call = exp(-r*T) * mean(finalC) ;
    put = exp(-r*T) * mean(finalP) ;
end
```

Listing 6: Matlab function that computes vanilla option prices with Monte Carlo using control variates variance reduction technique.

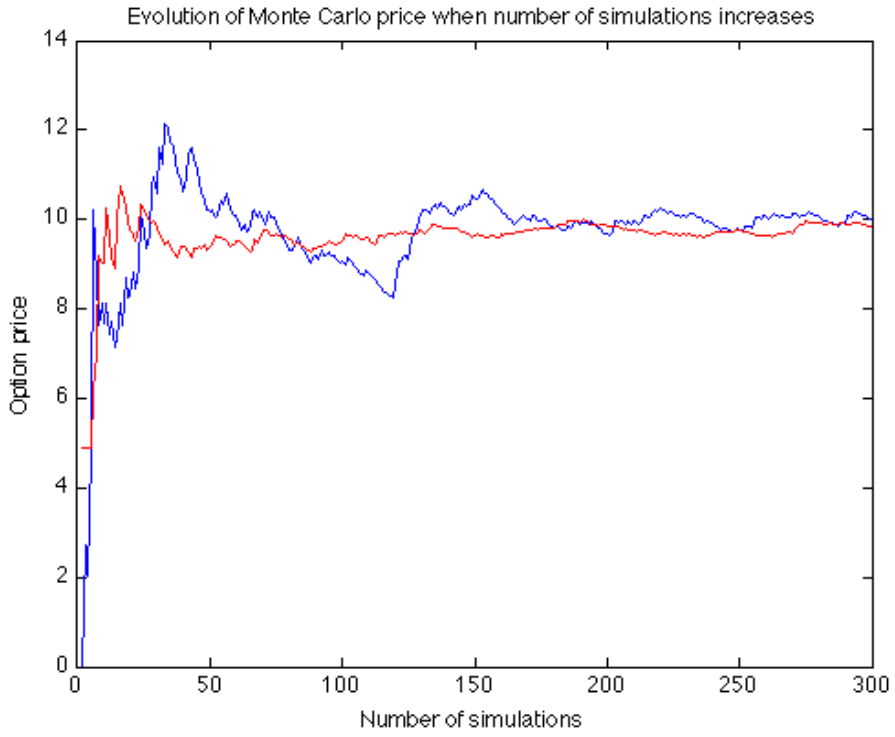

Figure 2: Evolution of the Monte Carlo option price when the number of simulations increases. Using control variates (red), we see the Monte Carlo price converges earlier.

4 Results

In Table 1 and Table 2, we show the results from a Monte Carlo simulation performed for both a call spread option and a put spread option, using the classic Monte Carlo method and the Monte Carlo method with antithetic variates variance reduction technique. We clearly see that the results using the antithetic variates converge to the Kirk approximation before. Indeed, it seems that the antithetic variates method needs five times less time steps in order to get the same confidence interval as the classic Monte Carlo method.

Kirk	9.0981	9.0981	9.0981	9.0981	9.0981	9.0981	9.0981
Step number	10	100	1,000	10,000	100,000	1,000,000	10,000,000
MC	9.3618	8.3274	9.4224	9.1567	9.1408	9.1021	9.1046
Confidence interval	107.1071	28.7563	9.1185	2.8510	0.9048	0.2841	0.0900
MC (antithetic)	8.8683	8.5975	9.0154	9.1327	9.0864	9.1020	9.0983
Confidence interval	14.2305	4.2395	1.6920	0.5669	0.1763	0.0559	0.0176

Table 1: Call option prices computed with the Kirk approximation, the classic Monte Carlo method and the Monte Carlo method with the antithetic variate variance reduction technique. The value of the parameters are $K = 5$, $S_1 = 90$, $S_2 = 80$, $\sigma_1 = 0.2$, $\sigma_2 = 0.2$, $\rho = 0.5$, $r = 0.05$, $T = 1$

Finally, in Figure 3 we show the confidence interval (which depends directly on the variance of the estimates) for vanilla option prices computed with the classic Monte Carlo method, the Monte Carlo method with antithetic variates and the Monte Carlo method with control variates. We clearly see that the estimates have a much lower variance when using a variance reduction technique, particularly the control variates one. Using a variance reduction technique, it requires much less simulation steps in order to get the same confidence interval as the classic Monte Carlo simulation one. The Monte Carlo method coupled with a variance reduction technique is thus much more efficient than the classic one.

Kirk	4.3420	4.3420	4.3420	4.3420	4.3420	4.3420	4.3420
Step number	10	100	1,000	10,000	100,000	1,000,000	10,000,000
MC	5.9389	4.3123	4.2061	4.3155	4.3218	4.3410	4.3383
Confidence interval	107.1071	28.7563	9.1185	2.8510	0.9048	0.2841	0.0900
MC (antithetic)	3.8329	3.6859	4.2429	4.3461	4.3220	4.3474	4.3433
Confidence interval	14.2305	4.2395	1.6920	0.5669	0.1763	0.0559	0.0176

Table 2: Call option prices computed with the Kirk approximation, the classic Monte Carlo method and the Monte Carlo method with the antithetic variate variance reduction technique. The value of the parameters are $K = 5$, $S_1 = 90$, $S_2 = 80$, $\sigma_1 = 0.2$, $\sigma_2 = 0.2$, $\rho = 0.5$, $r = 0.05$, $T = 1$

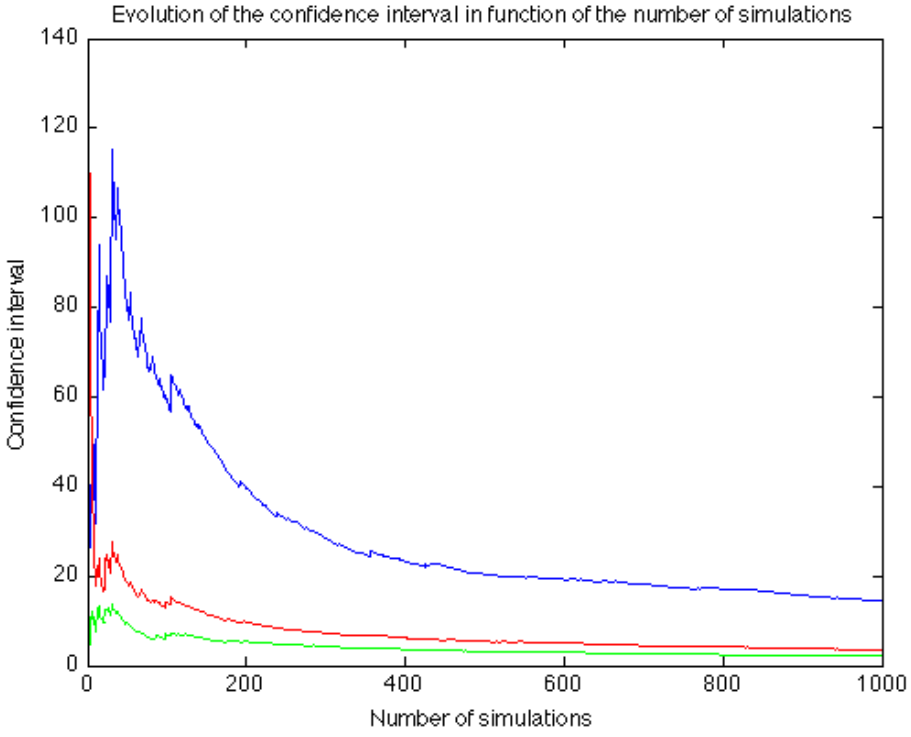

Figure 3: Evolution of the Monte Carlo simulation confidence interval value when the number of simulations increases. Using antithetic variates (red) and control variates (green) against the classic Monte Carlo (blue), we see that the variance of the estimates is lower.

5 Conclusion

We first presented the spread options and exposed their characteristics. We then explained the Monte Carlo framework for pricing vanilla options and showed how it could be extended in order to price exotic options, spread options in particular. We wrote some Matlab code that replicates these methods and compute option prices with the Monte Carlo framework. Finally, in order to increase the efficiency of the Monte Carlo method, we presented two variance reduction techniques: the antithetic variates and the control variates. We showed how these techniques actually decreased the variance of the estimates increasing radically the efficiency of the Monte Carlo method since much less steps were required in order to get the same result as a classic Monte Carlo method.

Some extensions to this project could be the introduction of a stochastic interest rate or a stochastic volatility in the model. We could also use other types of stochastic processes for the underlying such as processes displaying jumps as the Lévy processes.

Appendix

A Additional Matlab code

In this appendix, we list the Matlab code that has been used in order to generate the graphs.

```
function plotBS_MC(S,K,r,sigma,T,N,Z)
    %Z = randn(N,1) ;
    call = zeros(N,1) ;
    conf = zeros(N,1) ;
    for i=1:N
        S_fin = S * exp( (r-0.5*sigma^2)*T + sigma*sqrt(T)*Z(1:i,1)) ;
        resC = max(S_fin - K, 0) ;
        call(i) = exp(-r*T) * mean(resC) ;
        var = sum((resC-mean(resC)).^2)/(i) ;
        conf(i) = 1.96*var/sqrt(i) ;
    end
    plot(call) ;
    plot(conf) ;
end
```

Listing 7: Function that plot the call price and the confidence interval using the classic Monte Carlo method

.

```
function plotAntitheticBS_MC(S,K,r,sigma,T,N,Z)
    %Z = randn(N,1) ;
    nZ = -Z ;
    call = zeros(N,1) ;
    conf = zeros(N,1) ;
    for i=1:N
        pS_fin = S * exp((r-0.5*sigma^2)*T + sigma*sqrt(T)*Z(1:i,1)) ;
        nS_fin = S * exp((r-0.5*sigma^2)*T + sigma*sqrt(T)*nZ(1:i,1)) ;
        resCp = max(pS_fin - K, 0) ;
        resCn = max(nS_fin - K, 0) ;
        resC = 0.5 * (resCp + resCn) ;
        call(i) = exp(-r*T)*mean(resC) ;
        var = sum((resC-mean(resC)).^2)/(i) ;
        conf(i) = 1.96*var/sqrt(i) ;
```

```
15          end
16          plot(call,'r') ;
17          plot(conf,'r') ;
18      end
```

Listing 8: Function that plot the call price and the confidence interval using the Monte Carlo method with the antithetic variates

.

```
1       function plotControlBS_MC(S,K,r,sigma,T,N,Z)
2           %Z = randn(N,1) ;
3           call = zeros(N,1) ;
4           conf = zeros(N,1) ;
5           for i=1:N
6               S_fin = S*exp((r-0.5*sigma^2)*T + sigma*sqrt(T)*Z(1:i,1)) ;
7               resC = max(S_fin - K, 0) ;
8               newC = resC - mean(resC) ;
9               newS = S_fin - mean(S_fin) ;
10              bC = mean(newC.*newS)/mean(newS.*newS) ;
11              finalC = resC - bC*(S_fin - exp(r*T)*S) ;
12              call(i) = exp(-r*T) * mean(finalC) ;
13              var = sum((finalC-mean(finalC)).^2)/(i) ;
14              conf(i) = 1.96*var/sqrt(i) ;
15          end
16          plot(call,'g') ;
17          plot(conf,'g') ;
18      end
```

Listing 9: Function that plot the call price and the confidence interval using the Monte Carlo method with the control variates

References

[1] Andreas, A., Engelmann, B., Schwendner, P., and Wystup, U. Fast fourier method for the valuation of options on several correlated currencies. *Foreign Exchange Risk 2* (2002).

[2] Bjerksund, P., and Stensland, G. Closed form spread option valuation. *papers.ssrn.com* (2006).

[3] Black, F. The pricing of commodity contracts. *Journal of financial economics* (1976), 167–179.

[4] Boyle, P. P. A lattice framework for option pricing with two state variables. *Journal of Financial and Quantitative Analysis* (1988), 1–12.

[5] Carmona, R., and Durrleman, V. Pricing and hedging spread options. *Siam Review 45*, 4 (2003), 627–685.

[6] Carmona, R., and Durrleman, V. Pricing and hedging spread options in a lognormal model. *Siam Review 45*, 4 (2003), 627–685.

[7] Dempster, M., and Hong, S. Spread option valuation and the fast fourier transform. manuscript. *University of Cambridge* (2000).

[8] Duan, J., and Pliska, S. Option valuation with co-integrated asset prices. *Journal of Economic Dynamics and Control 28*, 4 (2004), 727–754.

[9] Geman, H. *Commodities and commodity derivatives*. Wiley, 2005.

[10] Glasserman, P. *Monte Carlo methods in financial engineering*. Springer, 2003.

[11] Margrabe, W. The value of an option to exchange one asset for another. *Journal of Finance* (1978), 177–186.

[12] Nakajima, K., and Maeda, A. Pricing commodity spread options with stochastic term structure of convenience yields and interest rates. *Asia-Pacific Finan Markets 14*, 1 (2007), 157–184.