Math 623 - Computational Finance

Double barrier option pricing using Quasi Monte Carlo and Brownian Bridge methods

Pratik Mehta

pbmehta@eden.rutgers.edu

Masters of Science in Mathematical Finance
Department of Mathematics, Rutgers University

This paper describes the implementation of a C++ program to calculate the value of a European style call option and a discretely sampled double barrier option on an underlying asset, given input parameters for stock price (s=100), strike price (k=110), volatility (v=30%), interest rate (r=5%), maturity (T=1 year), U = 120 knock in, and L = 90 knock out using Brownian bridge with terminal stratification, Quasi Monte Carlo simulation with low discrepancy Sobol sequences and Monte Carlo with antithetic sampling. We compare the results of all these methods. The underlying stock price is assumed to follow geometric brownian motion. The program calculates the option prices using the Monte Carlo/Quasi MC method with 10,000 simulations, 1000 strata for stratified sampling and uses the solution to the Stochastic differential equation of the stock process [2].

I. INTRODUCTION

In this assignment we are pricing a European style vanilla call option and discretely double barrier option (t = 6) using Brownian Bridge with terminal stratification, low discrepancy Sobol sequences and comparing results with closed form solution and Monte Carlo using antithetic sampling.

\[ dS(t) = (rS(t))dt + \sigma S(t)dW(t) \]  (1)

The solution to the above Stochastic Differential Equation is,

\[ S(T) = S(0)e^{(r-\frac{1}{2}\sigma^2)T+\sigma W(T)} \]  (2)

the payoff for the call option is,

\[ (S(T) - k)^+ \]  (3)

We also price a double barrier option, with the conditions, Up and in and Down and out. This can be represented by,

\[ (S(T) - k)^+ 1_{S(T) \geq U} 1_{S(T) \leq L} \]  (4)

We have modified Joshi’s code from chapter 7 (EquityFXMain.cpp), as was required for this assignment.

II. VANILLA CALL : CLOSED FORM BLACK-SCHOLES-MERTON

Here, K is the strike price and S(T) is the terminal stock price at the payoff date. The price of the underlying at terminal time T, is given by [3]

\[ S(T) = S(0)e^{(r-a-\frac{1}{2}\sigma^2)T+\sigma W(T)} \]  (5)

A closed form solution for the price of a European Call and Put is given by [3]

\[ c(t, x) = xe^{-aT}N(d_+(T, x)) - e^{-rT}KN(d_-(T, x)) \]  (6)

and

\[ p(t, x) = e^{-rT}KN(-d_-(T, x)) - xe^{-aT}N(-d_+(T, x)) \]  (7)

Here, N is the Normal Cumulative Distribution density and the parameters, \( d_+ \) and \( d_- \) are given by

\[ d_+ = d_- + \sigma \sqrt{T} = \frac{1}{\sigma \sqrt{T}} \left[ \log \frac{x}{K} + \left( r + \frac{1}{2} \sigma^2 \right) T \right] \]  (8)

The function ”bsm” in the file ”EquityMainFX.cpp” describes the implementation of the above closed form Black Scholes Merton model. The output is as follows,

\[ \text{Closed form option price} = 9.057 \]  (9)

III. VANILLA CALL : MONTE CARLO WITH PARK-MILLER AND ANTITHETIC SAMPLING

Park-Miller is one of the many random number generators that are used to produce random samples for Monte Carlo simulation. It is one of the Linear Congruential Generators (LCG). It has a period of,

\[ 2^{31} - 1 \]  (10)

The general form of a LCG generator is,

\[ x_{i+1} = (ax_i + c) \mod m_1 \]
\[ u_{i+1} = x_{i+1} / m_1 \]  (11)
x = initial seed
a = multiplier
m = modulus
c = shift

The following are the values for the variables

\[ m = 2^{31} - 1, a = 16807 \] (13)

Park Miller’s uniform random number generator is implemented by Mark Joshi. We then pass these uniform values to inverse cumulative distribution to get normally distributed variables. We use antithetic sampling which is described below. We use the barrier class to price the vanilla call option. We set the Upper barrier \( U = 100 \) and \( L = 0 \), so that the barrier option now behaves like a vanilla option. We have implemented this function in the file BrownianBridgePath.h and cpp. In Antithetic sampling we take one sample using Park Miller and the use the negative of the same value as second sample. In this was we get the following equation

\[ S(t_{i+1}) = S(t_i)(1 + r\tau + \sigma\sqrt{T}Z_{i+1}) \] (14)

\[ S(t_{i+1}) = S(t_i)(1 + r\tau - \sigma\sqrt{T}Z_{i+1}) \] (15)

Using the above formula, we can reduce the variance of the samples. The result of Monte Carlo simulation on using the above variance reduction technique.

\[ \hat{Y}_{AV} = \frac{1}{n}(\sum_{i=1}^{n}\frac{\tilde{Y}_i + Y_i}{2}) \] (16)

The output for out call option is,

\[ MC \text{ vanilla call with Park} \text{- Miller} = 9.036 \] (17)

IV. DOUBLE BARRIER OPTIONS

Barrier options are path-dependent options, with payoffs that depend on the price of the underlying asset at expiration and whether or not the asset price crosses a barrier during the life of the option. There are two categories or types of Barrier options: "knock-in" and "knock-out". "Knock-in" or "in" options are paid for up front, but you do not receive the option until the asset price crosses the barrier. This Knock-in can be represented by,

\[ (S(T) - k)^+1_{\max(S(t)) \geq U} \] (18)

"Knock-out" or "out" options come into existence on the issue date but becomes worthless if the asset price hits the barrier before the expiration date. If the option is a knock-in (knock-out), a predetermined cash rebate may be paid at expiration if the option has not been knocked in (knocked-out) during its lifetime. The barrier monitoring frequency specifies how often the price is checked for a breach of the barrier. All of the analytical models have a flag to change the monitoring frequency where the default frequency is continuous. Knock-out options are represented as

\[ (S(T) - k)^+1_{S(t) \geq L} \] (19)

Double Barrier options are options with two barriers. In this assignment we price a boucle barrier option with knock in upper barrier \( U \) and a knock out lower barrier \( L \). \( U = 120, L = 90 \). This means the option will only pay if we cross \( U \) at least once on the discrete monitoring dates and always remain above the barrier \( L \).[8] This option can be represented as,

\[ (S(T) - k)^+1_{\max(S(t)) \geq U}1_{S(t) \geq L} \] (20)

V. DOUBLE BARRIER WITH PARK - MILLER UNIFORMS AND ANTITHETIC SAMPLING

Park Miller’s uniform random number generator is implemented by Mark Joshi. We then pass these uniform values to inverse cumulative distribution to get normally distributed variables. We use antithetic sampling which is described below. We use the barrier class to price double barrier call option. We set the Upper barrier \( U = 120 \) and \( L = 90 \). We have implemented this function in the file BrownianBridgePath.h and cpp. In Antithetic sampling we take one sample using Park Miller and the use the negative of the same value as second sample. In this was we get the following equation

\[ S(t_{i+1}) = S(t_i)(1 + r\tau + \sigma\sqrt{T}Z_{i+1}) \] (21)

\[ S(t_{i+1}) = S(t_i)(1 + r\tau - \sigma\sqrt{T}Z_{i+1}) \] (22)

We use the above formula for path generation and then check for the barrier conditions, and if we satisfy the conditions than we calculate the payoff and then average them.

Using the above formula, we can reduce the variance of the samples. The result of Monte Carlo simulation on using the above variance reduction technique.

\[ \hat{Y}_{AV} = \frac{1}{n}(\sum_{i=1}^{n}\frac{\tilde{Y}_i + Y_i}{2}) \] (23)

\[ MC \text{ double barrier Park} \text{- Miller antithetics} = 8.6 \] (24)
VI. BROWNIAN BRIDGE WITH TERMINAL STRATIFICATION - VANILLA CALL AND DOUBLE BARRIER OPTIONS

The brownian bridge path generation is usually used in conjunction with variance reduction techniques like stratified sampling. Suppose \( W(t) \) is a \( R \)-valued Brownian motion and suppose we know the some values of Brownian motion,

\[
W(s_1) = x_1, \ldots, W(s_k) = x_k
\]

we can find the value of \( W(s) \) where \( s \) in \((s_i, s_{i+1})\) conditional on the \( k \) values. This conditional distribution of \( W(s) \) is on

\[
W(s_i) = x_i
\]

\[
W(s_{i+1}) = x_{i+1}
\]

and is given by,

\[
W(s) = \frac{(s_{i+1} - s_i)W(s_i) + (s - s_i)W(s_{i+1})}{(s_{i+1} - s_i)} + \sqrt{\frac{(s_{i+1} - s_i)(s - s_i)}{(s_{i+1} - s_i)}}
\]

where \( Z \) is Normally distributed random variable.

**Stratified (Regional) sampling.**

Stratified sampling is a divide-and-conquer strategy which partitions the state space into regions, and then a sampling plan is devised for each region. Moreover, common sense suggests that the smaller the region, the less variance that should exist in that region, and so a variance reduction may be achievable in each of the smaller regions that partition the entire space.

Divide \( U \) into \( k \) subdivisions, or *strata*, \( U_i \), where \( k \) is the total number of possible combinations described above. We have

\[
U = U_1 \cup U_2 \cup \ldots \cup U_k \text{ such that } U_i \cap U_j = \emptyset,
\]

for all \( i \neq j \) and \( 1 \leq i, j \leq k \). if we let \( N_i = U_i \), then

\[
N = \sum_{i=1}^{k} N_i.
\]

Draw a sample \( S_i \) from each stratum \( U_i \). The above description of asian option was taken from [6] The algorithm for implementing Brownian bridge with Stratification is given below

\[
W(t_m) = \sqrt{t_m} \phi^{-1}
\]

\[
a = \frac{(s_{i+1} - s_i)W(s_i) + (s - s_i)W(s_{i+1})}{(s_{i+1} - s_i)}
\]

\[
b = \sqrt{\frac{(s_{i+1} - s_i)(s - s_i)}{(s_{i+1} - s_i)}}
\]

\[
W_i = a + bZ
\]

\[
return(W(t_1) ... W(t_m))
\]

We have implemented this algorithm in the file QuasiMonteCarloPath.h and QuasiMonteCarloPath.cpp. It is implemented as a class and uses different member function to calculate prices of Vanilla and Asian options. We have used the above algorithm in conjunction with Park-Miller uniforms.

For vanilla Asian call options

\[
MC \text{ Vanilla, Park-Miller, stratified} = 8.91
\]

For Double barrier call options

\[
MC \text{ Double Barrier stratified} = 7.86
\]

VII. DOUBLE BARRIER/VANILLA CALL, QUASI MONTE CARLO WITH SOBOL SEQUENCE

A major drawback of Monte Carlo integration with pseudo-random numbers is given by the fact that the error term scales only as \( 1/\sqrt{N} \). This is inherent to methods based on random numbers. This leads to the idea to choose the points deterministically such that to minimize the integration error. Low-discrepancy sequence is a sequence with the property that for all values of \( N \), its subsequence \( x_1, ..., x_N \) has a low discrepancy. Roughly speaking, the discrepancy of a sequence is low if the number of points in the sequence falling into an arbitrary set \( B \) is close to proportional to the measure of \( B \), as would happen on average (but not for particular samples) in the case of a uniform distribution[4].

We now have the main theorem of quasi-Monte Carlo integration: If \( f \) has bounded variation on \([0, 1]^d\) then for any \( x_1, ..., x_N \in [0, 1]^d \) we have

\[
\left| \frac{1}{N} \sum_{n=1}^{N} f(x_n) - \int \! dx \! f(x) \right| \leq V(f)D^*(x_1, ..., x_N).
\]

Sobol sequences [5] are obtained by first choosing a primitive polynomial over \( Z \) of degree \( g \):

\[
P = x^g + a_1 x^{g-1} + .. + a_g x + 1,
\]
where each \(a_i\) is either 0 or 1. The coefficients \(a_i\) are used in the recurrence relation

\[v_i = a_1v_{i-1} \oplus a_2v_{i-2} \oplus ... \oplus a_{g-1}v_{i-g+1} \oplus v_{i-g} \oplus [v_{i-g}(2^g)]\]

where \(\oplus\) denotes the bitwise exclusive-or operation and each \(v_i\) is a number which can be written as \(v_i = m_i/(2^i)\) with \(0 < m_i < 2^i\). Consecutive quasi-random numbers are then obtained from the relation

\[x_{n+1} = x_n \oplus v_c,\]  

(46)

where the index \(c\) is equal to the place of the rightmost zero-bit in the binary representation of \(n\). For example \(n = 11\) has the binary representation 1011 and the rightmost zero-bit is the third one. Therefore \(c = 3\) in this case. For a primitive polynomial of degree \(g\) we also have to choose the first \(g\) values for the \(m_i\). The only restriction is that the \(m_i\) are all odd and \(m_i < 2^i\). There is an alternative way of computing \(x_n\):

\[x_n = g_1v_1 \oplus g_2v_2 \oplus ...\]

(47)

where \(g_3g_2g_1\) is the Gray code representation of \(n\). The basic property of the Gray code is that the representations for \(n\) and \(n + 1\) differ in only one position. The Gray code representation can be obtained from the binary representation according to

\[...g_3g_2g_1 = ...b_3b_2b_1 \oplus ...b_4b_3b_2.\]  

(48)

The above description of the sobol sequence generation was taken from [1]. We have used Sobol sequence generator by S.Joe and R.Y.Kuo. We pass a file with the primitive polynomial and values of all the coefficient. This Sobol generator is capable of producing vary high dimensional sequences from 0 to 1. In our application we need only 1 dimensional sequence since we are evaluating a vanilla call option at expiry. We have implemented a function 

\texttt{MCSobol in Sobol.cpp}

which uses sobol sequences to implement Quasi Monte Carlo simulation. The function "\texttt{sobolpoints}" returns an array with sobol sequences which are used for option price calculation. The price with this method is,

\[\text{QMC vanilla call with Sobol sequences} = 9.056\]  

(49)

The double barrier call option with Sobol sequence price is,

\[\text{QMC Double Barrier Sobol sequences} = 8.10\]  

(50)

As we can see that the option price using sobol sequences and quasi monte carlo method is the closest to the closed formed solution.

\section{VIII. ANSWER TO THE QUESTIONS}

\textbf{Q1}

We have compared our results with those of the closed-form continuously monitored single barrier options from Kerry Back’s and Espen Haug’s spreadsheets. We have attached the snapshots below. The results are as follows,

\[\text{Haug down and out call} = 6.335\]  

(51)

\[\text{Haug up and in call} = 9.006\]  

(52)

\[\text{Black's down and out call} = 6.33\]  

(53)

\section{Q2}

We have compared our results with those of the closed-form continuously monitored double barrier options from Espen Haug’s spreadsheet. We have compared our results,

\[\text{Haug up – in and down – in call} = 9.05\]  

(54)

\[\text{Haug up – out and down – out call} = 0.0035\]  

(55)

\section{Q3 Brownian Bridge}

\textbf{Effect of varying strata size}

We have varied the number of strata, and hence the strata size and calculated the option price. We used the following number of strata, 10,100,300,600,900,1200 and 5000. The results of varying the number of strata are presented in the second table below. We find that varying the size changes the answers marginally. The change is not significant. The answer seems to increase with the strata size till about 600 and then decreases.

\section{IX. BENCHMARKING}

We have used many spreadsheet based model to benchmark our results. The summary go the benchmarking is given below and is also tabularised below. Excel spreadsheets by Haug and Black does not provide \textbf{Up-and-in-down-and-out call option}. Hence we have benchmarked our results using, Numerix, the closed form single barrier and up and in down and in double barrier options from the VBA sheets. The results are,

\[\text{Numerix option price vanilla} = 9.08\]  

(56)

\[\text{Numerix up – in – and – down – out} = 8.24\]  

(57)

\[\text{Haug down and out call} = 6.335\]  

(58)

\[\text{Haug up and in call} = 9.006\]  

(59)

\[\text{Black's down and out call} = 6.33\]  

(60)

\[\text{Haug up – in and down – in call} = 9.05\]  

(61)

\[\text{Haug up – out and down – out call} = 0.0035\]  

(62)
### TABLE I: Vanilla and Asian Call pricing using Methods for $S(0)=100$ and $K=110$

<table>
<thead>
<tr>
<th>Option</th>
<th>Closed Form</th>
<th>Park-Miller Antithetic</th>
<th>Park-Miller with Stratified Sobol sequence</th>
<th>Numerix</th>
</tr>
</thead>
<tbody>
<tr>
<td>European Vanilla Call</td>
<td>9.05</td>
<td>9.036</td>
<td>8.91</td>
<td>9.056</td>
</tr>
<tr>
<td>Double barrier (Up-in, Down-out) Call</td>
<td>-</td>
<td>8.60</td>
<td>7.86</td>
<td>8.10</td>
</tr>
</tbody>
</table>

### TABLE II: Effect of varying Strata size $S(0)=100$ and $K=110$

<table>
<thead>
<tr>
<th>Strata size</th>
<th>Double barrier call</th>
<th>Vanilla Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>7.79</td>
<td>8.81</td>
</tr>
<tr>
<td>100</td>
<td>7.82</td>
<td>8.83</td>
</tr>
<tr>
<td>300</td>
<td>7.83</td>
<td>8.82</td>
</tr>
<tr>
<td>600</td>
<td>7.78</td>
<td>8.78</td>
</tr>
<tr>
<td>900</td>
<td>7.84</td>
<td>8.81</td>
</tr>
<tr>
<td>1200</td>
<td>7.80</td>
<td>8.79</td>
</tr>
<tr>
<td>5000</td>
<td>7.80</td>
<td>8.78</td>
</tr>
</tbody>
</table>
### Standard Barrier option

Implementation By Espen Gaarder Haug, Copyright 2006


<table>
<thead>
<tr>
<th>Time in:</th>
<th>Years</th>
<th>Long</th>
<th>Continuously</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asset price (S)</td>
<td>100.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Strike price (X)</td>
<td>110.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Barrier (H)</td>
<td>90.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rebate (K)</td>
<td>0.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time to maturity (T)</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Risk-free rate (r)</td>
<td>5.00%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cost of carry (b)</td>
<td>3.00%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Volatility (σ)</td>
<td>30.00%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adjusted Barrier</td>
<td>90.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Forward price</td>
<td>103.0455</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Value</td>
<td>6.3350</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

By Espen Gaarder Haug

FIG. 1: Haug, Down-and-Out.

### Standard Barrier option

Implementation By Espen Gaarder Haug, Copyright 2006


<table>
<thead>
<tr>
<th>Time in:</th>
<th>Years</th>
<th>Long</th>
<th>Continuously</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asset price (S)</td>
<td>100.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Strike price (X)</td>
<td>110.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Barrier (H)</td>
<td>120.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rebate (K)</td>
<td>0.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time to maturity (T)</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Risk-free rate (r)</td>
<td>5.00%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cost of carry (b)</td>
<td>3.00%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Volatility (σ)</td>
<td>30.00%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adjusted Barrier</td>
<td>120.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Forward price</td>
<td>103.0455</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Value</td>
<td>9.0063</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

By Espen Gaarder Haug

FIG. 2: Haug, Up-and-In.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asset price S</td>
<td>100.00</td>
</tr>
<tr>
<td>Exercise price K</td>
<td>110.00</td>
</tr>
<tr>
<td>Interest rate r</td>
<td>0.05</td>
</tr>
<tr>
<td>Volatility sigma</td>
<td>0.3</td>
</tr>
<tr>
<td>Dividend yield q</td>
<td>0.02</td>
</tr>
<tr>
<td>Time to maturity T</td>
<td>1</td>
</tr>
<tr>
<td>Barrier</td>
<td>90</td>
</tr>
<tr>
<td><strong>Down_And_Out_Call</strong></td>
<td>6.334982</td>
</tr>
</tbody>
</table>

**FIG. 3**: Black, Down-and-Out.

---

**2. DoubleBarrier**

**Double Barrier Option (Ikeda Kuntimo) only holds w**

Implementation By Espen Gaarder Haug, Copyright 2006

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asset price (S)</td>
<td>100.00</td>
</tr>
<tr>
<td>Strike price (X)</td>
<td>110.00</td>
</tr>
<tr>
<td>Lower (L)</td>
<td>90.00</td>
</tr>
<tr>
<td>Upper (U)</td>
<td>120.00</td>
</tr>
<tr>
<td>Time to maturity (T)</td>
<td>1.00</td>
</tr>
<tr>
<td>Risk-free rate (r)</td>
<td>5.00%</td>
</tr>
<tr>
<td>Cost of carry (b)</td>
<td>3.00%</td>
</tr>
<tr>
<td>Volatility (σ)</td>
<td>30.00%</td>
</tr>
<tr>
<td>Upper curvature (δ₁)</td>
<td>0.00</td>
</tr>
<tr>
<td>Lower curvature (δ₂)</td>
<td>0.00</td>
</tr>
<tr>
<td>Adjusted lower barrier</td>
<td>90.00000</td>
</tr>
<tr>
<td>Adjusted upper barrier</td>
<td>120.00000</td>
</tr>
<tr>
<td>Forward price</td>
<td>103.04545</td>
</tr>
<tr>
<td>Value</td>
<td>9.0535</td>
</tr>
</tbody>
</table>

By Espen Gaarder Haug

**FIG. 4**: Haug Double Barrier, Up-and-In, Down-and-in
Double Barrier Option (Ikeda Kuntimo) only holds when $X$ is inside barrier range, for $X$ outside barrier range look into Double Barrier Symmetry by Haug.

Implementation By Espen Gaarder Haug, Copyright 2006

FIG. 5: Numerix, Double Barrier

2.DoubleBarrier

Double Barrier Option (Ikeda Kuntimo) only holds when $X$ is inside barrier range, for $X$ outside barrier range look into Double Barrier Symmetry by Haug.

Implementation By Espen Gaarder Haug, Copyright 2006

Time in:

- Years
- Long
- Continuously

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asset price</td>
<td>100.00</td>
</tr>
<tr>
<td>Strike price</td>
<td>110.00</td>
</tr>
<tr>
<td>Lower</td>
<td>90.00</td>
</tr>
<tr>
<td>Upper</td>
<td>120.00</td>
</tr>
<tr>
<td>Time to maturity</td>
<td>1.00</td>
</tr>
<tr>
<td>Risk-free rate</td>
<td>5.00 %</td>
</tr>
<tr>
<td>Cost of carry</td>
<td>3.00 %</td>
</tr>
<tr>
<td>Volatility</td>
<td>30.00 %</td>
</tr>
<tr>
<td>Upper curvature</td>
<td>0.00</td>
</tr>
<tr>
<td>Lower curvature</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Adjusted lower barrier: 90.00000
Adjusted upper barrier: 120.00000
Forward price: 103.04545

Value: 0.0035

By Espen Gaarder Haug

FIG. 6: Haug Double Barrier, Up-and-out, Down-and-out
[1] www.thep.physik.uni-mainz.de/ste-fanw/download/lecture
[8] rss.aces.unt.edu/Rdoc/library/fExoticOptions/latex/BarrierOptions