15
Double Barrier Options

A double barrier option is knocked out or in if the asset price touches a lower (L) or upper (U) barrier level within the option lifetime. Double barrier option valuation has been described by Geman & Yor (1996), Kunitomo & Ikeda (1992) and others. The approach taken by Geman & Yor is a probabilistic model which uses the Laplace transform of the option price. The pricing formulae of Kunitomo & Ikeda take an approach based on the pricing being constrained by curved boundaries, this approach therefore has the advantage of covering barriers which are flat or have exponential growth/decay or are concave. We will develop classes for the Kunitomo & Ikeda formulae and a simpler set of methods based on the work of Haug (1999), which gives a set of methods for pricing barrier options with flat boundaries. The methods based on Haug have the advantage of using standard (plain vanilla) barrier options to construct a double barrier pricing mechanism.

15.1. Double Knock In/Out

15.1.1. Double Knock Out Call

The price of a double knock-out call has payoff:max$[S_T - K; 0]$; $L < S_\tau < U$: *else*, 0. Thus, if the underlying asset remains above the lower barrier and below the upper barrier until maturity, the payoff is the standard Black Scholes call payoff. If the asset price touches either the upper or lower boundary, then the option is knocked out worthless (zero payoff).

The formula for the double knock-out call is:

$$\begin{aligned} c = Se^{(b-r)T} \sum_{n=-\infty}^{\infty} \Bigg\{ \left(\frac{U^n}{L^n}\right)^{\mu_1} \left(\frac{L}{S}\right)^{\mu_2} [\phi(d_1) - \phi(d_2)] \\ - \left(\frac{L^{n+1}}{U^n S}\right)^{\mu_3} [\phi(d_3) - \phi(d_4)] \Bigg\} \\ - Xe^{-rT} \sum_{n=-\infty}^{\infty} \Bigg\{ \left(\frac{U^n}{L^n}\right)^{\mu_1 - 2} \left(\frac{L}{S}\right)^{\mu_2} [\phi(d_1 - \sigma\sqrt{T}) - \phi(d_2 - \sigma\sqrt{T})] \\ - \left(\frac{L^{n+1}}{U^n S}\right)^{\mu_3 - 2} [\phi(d_3 - \sigma\sqrt{T}) - \phi(d_4 - \sigma\sqrt{T})] \Bigg\} \end{aligned} \qquad (15.1.1)$$

Where

$$d_1 = \frac{\log(SU^{2n} \ / \ XL^{2n}) + (b + \sigma^2 \ / \ 2)T}{\sigma\sqrt{T}}$$

$$d_2 = \frac{\log(SU^{2n} \ / \ FL^{2n}) + (b + \sigma^2 \ 2)T}{\sigma\sqrt{T}}$$

$$d_3 = \frac{\log(L^{2n+2} \ / \ XSU^{2n}) + (b + \sigma^2 \ / \ 2)T}{\sigma\sqrt{T}}$$

$$d_4 = \frac{\log(L^{2n+2}/FSU^{2n}) + (b + \sigma^2 \ / \ 2)T}{\sigma\sqrt{T}}$$

$$\mu_1 = \frac{2[b - \delta_2 - n(\delta_1 - \delta_2)]}{\sigma^2} + 1$$

$$\mu_2 = 2n\frac{(\delta_1 - \delta_2)}{\sigma^2}$$

$$\mu_3 = \frac{2[b - \delta_2 + n(\delta_1 - \delta_2)]}{\sigma^2} + 1, F = Ue^{\delta_1 T}$$

The variables δ_1, δ_2 represent the boundary curvature. For $\delta_1 = \delta_2 = 0$, the boundaries are flat.

For $\delta_1 < 0 < \delta_2$, the lower boundary exhibits exponential growth and the upper boundary exhibits exponential decay. The case of $\delta_1 > 0 > \delta_2$ is a convex lower boundary and convex upper boundary.

The double barrier knock-out option can be represented as a portfolio of a standard option and a double barrier knock-in. The resultant characteristics are shown in Figure 15.1. At each point on the graph we can see that the value of an up and-out down and-out call option is given by the Black Scholes standard call value minus the value of the up and-in down and-in call option.

With

$$S = 100.0, X = 100.0, \delta_1 = \delta_2 = 0.0, L = 80.0, U = 120.0, T = 0.25.$$

Table 15.1 shows a range of valuations for up and-out down and-out calls, with various curve boundary parameters.

Table 15.2 shows an example range of valuations for call up and-in down and-in barrier calls.

15.1.2. Double Knock Out Put

The price of a double knock-out put has payoff:max$[X - S; 0], L < S < U; else, 0$. Thus, if the underlying asset remains above the lower barrier and below the upper barrier until maturity, the payoff is the standard Black Scholes put payoff. If the asset price touches either the upper or lower boundary, then the option is knocked out.

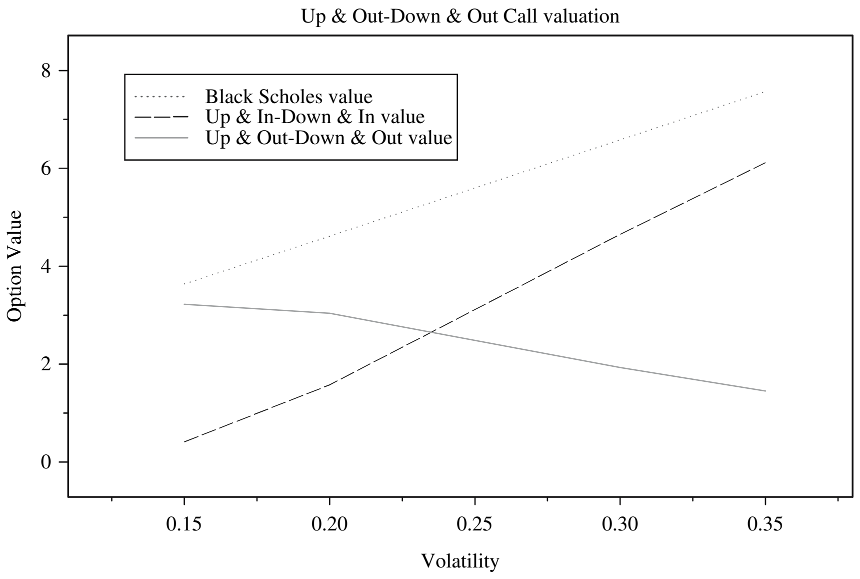

FIGURE 15.1. Up & Out-Down & Out Call barrier option as a portfolio of a long vanilla call and short an Up & In-Down & In barrier call.

TABLE 15.1. Call Up & Out-Down & Out valuations

S=110.0, X=110.0 r=0.05, b=0.05				Time 0.25				Time 0.5			
				σ1	σ2	σ3	σ4	σ1	σ2	σ3	σ4
U	L	δ1	δ2	0.10	0.15	0.25	0.35	0.10	0.15	0.25	0.35
150.0	60.0	0.0	0.0	2.9313	3.9962	5.5857	5.3532	4.6097	5.8498	5.5975	3.795
140.0	70.0	0.0	0.0	2.9312	3.9401	4.4224	3.3301	4.557	5.0486	3.4107	1.8949
130.0	80.0	0.0	0.0	2.8968	3.308	2.2613	1.2561	3.8655	2.955	1.2562	0.5492
120.0	90.0	0.0	0.0	1.7046	0.9845	0.323	0.1164	1.1608	0.4927	0.1125	0.0184
110.0	100.0	0.00	0.00	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
U	L	−δ1	δ2								
150.0	60.0	0.05	0.05	2.9313	3.9942	5.4568	5.0659	4.6051	5.6815	4.9531	3.1875
140.0	70.0	0.05	0.05	2.931	3.9065	4.1546	3.0139	4.4708	4.5591	2.7475	1.4537
130.0	80.0	0.05	0.05	2.8626	3.0847	1.9411	1.0408	3.2969	2.2022	0.8387	0.3451
120.0	90.0	0.05	0.05	1.3268	0.6932	0.2116	0.0711	0.5197	0.1981	0.0372	0.0042
110.0	100.0	0.05	0.05	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
U	L	δ1	−δ2								
150.0	60.0	0.05	0.05	2.9313	3.9973	5.6952	5.6271	4.6111	5.9536	6.191	4.4221
140.0	70.0	0.05	0.05	2.9313	3.9624	4.6674	3.6468	4.5927	5.414	4.0844	2.3852
130.0	80.0	0.05	0.05	2.9149	3.4894	2.5857	1.4887	4.2324	3.6727	1.7486	0.8071
120.0	90.0	0.05	0.05	2.041	1.3056	0.4612	0.1763	1.9468	0.9316	0.2454	0.0521
110.0	100.0	0.05	0.05	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

TABLE 15.2. Call Up & In-Down & In valuations

S=110.0, X=110.0 r=0.05, b=0.05				Time 0.25				Time 0.5			
				σ1	σ2	σ3	σ4	σ1	σ2	σ3	σ4
U	L	δ1	δ2	0.10	0.15	0.25	0.35	0.10	0.15	0.25	0.35
150.0	60.0	0.0	0.0	0.000	0.0024	0.5725	2.9716	0.0017	0.230	3.4885	8.3165
140.0	70.0	0.0	0.0	0.0001	0.0585	1.7359	4.9947	0.0545	1.0313	5.6753	10.2166
130.0	80.0	0.0	0.0	0.0346	0.6906	3.8969	7.0688	0.746	3.1248	7.8299	11.5623
120.0	90.0	0.0	0.0	1.2268	3.014	5.8352	8.2084	3.4507	5.5871	8.9735	12.0931
110.0	100.0	0.0	0.0	2.9313	3.9986	6.1582	8.3248	4.6115	6.0798	9.086	12.1115
U	L	−δ1	δ2								
150.0	60.0	0.05	0.05	0.000	0.0044	0.7014	3.2589	0.0064	0.3983	4.1329	8.924
140.0	70.0	0.05	0.05	0.0004	0.0921	2.0036	5.3109	0.1407	1.5207	6.3385	10.6578
130.0	80.0	0.05	0.05	0.0687	0.9139	4.2172	7.284	1.3146	3.8776	8.2473	11.7663
120.0	90.0	0.05	0.05	1.6045	3.3053	5.9466	8.2537	4.0918	5.8817	9.0489	12.1073
110.0	100.0	0.05	0.05	2.9313	3.9986	6.1582	8.3248	4.6115	6.0798	9.086	12.1115
U	L	δ1	−δ2								
150.0	60.0	0.05	0.05	0.000	0.0013	0.4631	2.6977	0.0004	0.1262	2.895	7.6894
140.0	70.0	0.05	0.05	0.000	0.0362	1.4908	4.678	0.0188	0.6658	5.0016	9.7262
130.0	80.0	0.05	0.05	0.0164	0.5092	3.5725	6.8362	0.3791	2.4071	7.3374	11.3044
120.0	90.0	0.05	0.05	0.8903	2.693	5.6971	8.1485	2.6647	5.1482	8.8406	12.0594
110.0	100.0	0.05	0.05	2.9313	3.9986	6.1582	8.3248	4.6115	6.0798	9.086	12.1115

The Kunitomo & Ikeda formula for an up and-out down and-out put is:

$$p = Xe^{-rT} \sum_{n=-\infty}^{n=\infty} \left\{ \left(\frac{U^n}{L^n}\right)^{\mu_1 - 2} \left(\frac{L}{S}\right)^{\mu_2} [\phi(y_1 - \sigma\sqrt{T}) - \phi(y_2 - \sigma\sqrt{T})] - \left(\frac{L^{n+1}}{U^n S}\right)^{\mu_3 - 2} [\phi(y_3 - \sigma\sqrt{T}) - \phi(y_4 - \sigma\sqrt{T})] \right\}$$

$$- Se^{(b-r)T} \sum_{n=-\infty}^{n=\infty} \left\{ \left(\frac{U^n}{L^n}\right)^{\mu_1} \left(\frac{L}{S}\right)^{\mu_2} [\phi(y_1) - \phi(y_2)] - \left(\frac{L^{n+1}}{U^n S}\right)^{\mu_3} [\phi(y_3) - \phi(y_4)] \right\} \tag{15.1.2}$$

Where

$$y_1 = \frac{\log(SU^{2n} \ / \ EL^{2n}) + (b + \sigma^2 \ / \ 2)T}{\sigma\sqrt{T}}$$

$$y_2 = \frac{\log(SU^{2n} \ / \ XL^{2n}) + (b + \sigma^2 \ / \ 2)T}{\sigma\sqrt{T}}$$

$$y_3 = \frac{\log(L^{2n+2} \ / \ ESU^{2n}) + (b + \sigma^2 \ / \ 2)T}{\sigma\sqrt{T}}$$

$$y_3 = \frac{\log(L^{2n+2} \ / \ XSU^{2n}) + (b + \sigma^2/2)T}{\sigma\sqrt{T}}, E = Le^{\delta_1 T}$$

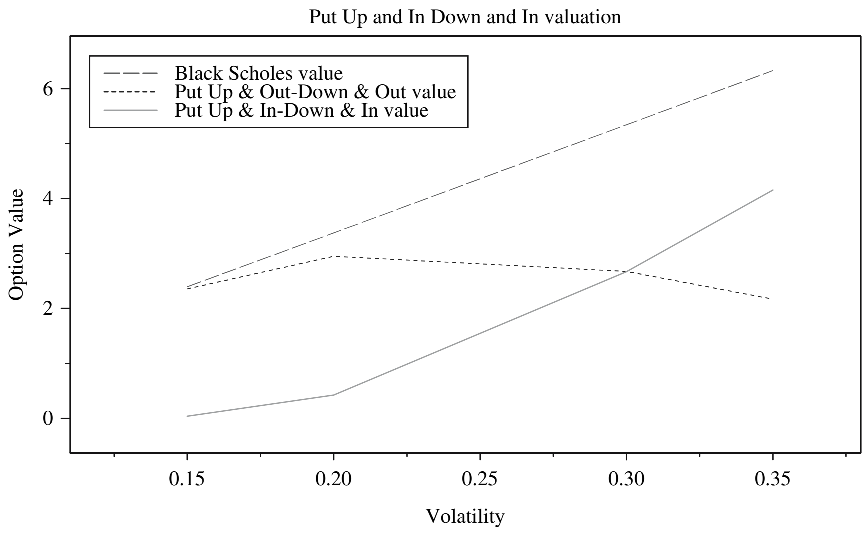

FIGURE 15.2. Up & In-Down & In barrier option as a portfolio of a long vanilla put and short an Up & Out-Down & Out barrier put.

15.1.3. Double Knock In Put

The up and-in down and-in barrier put option can be represented by a portfolio of long a standard Black-Scholes put option and a short an up and-out down and-out put. The resultant characteristic is shown in Figure 15.2.

The parameters for Figure 15.2. are:

$$S = 100.0, X = 100.0, \delta_1 = \delta_2 = 0.0, L = 80.0, U = 120.0, T = 0.25.$$

Tables 15.3. and 15.4. show example values for the remaining two double barrier options, the put up and-in down and-in, together with the put up and-out down and-out.

Double Barrier pricing methods are in the class ***Dbarrierop*** shown in Listing 15.1.

```
package FinApps;
import static java.lang.Math.*;
import BaseStats.Probnorm;
public class Dbarrierop {
    /** Creates a new instance of Dbarrierop */
    public Dbarrierop(double rate,double q,double time, double volatility,
        int period ) {
        crate=q;
        r=rate;
        t=time;
        //0 continuos,1 hourly,2 daily,3 weekly, 4 monthly
```

```
      sigma=volatility;
      tau=period==1?(1.0/(24*365)):period==2?(1.0/(365.0)):
                  period==3?(1.0/(52.0)):period==4?(1.0/(12.0)):0.0;
                  b=crate==0.0?0.0:(b=crate!=r?(r-crate):r);
   }
   double b;
   double tau;
   double crate;
   double t;
   double r;
   double s;
   double u;
   double x;
   double l;
   double sigma;
   double f;
   double delta1;
   double delta2;
   public double uoDoc(double stock, double strike, double up,
                     double low,double delt1, double delt2 )
   {
      s=stock;
      x=strike;
      u= up>s? (up*exp(sqrt(tau)*sigma*0.5826)):up;
      l= low<s? (low*exp(-sqrt(tau)*sigma*0.5826)):low;
         if(s>=u|s<=l)
         return 0.0;// no need to continue
      delta1=delt1;
      delta2=delt2;
      double mu2;
      int n=0;
      double sum=0.0;
      double sum1=0.0;
      double c;
      f=(u*exp(delta1*t));
      for(n=-5;n<6;n++)
      {
         mu2=2.0*n*((delta1-delta2)/(sigma*sigma));
         sum+=pow((pow(u,n)/pow(l,n)),mu1(n)) *pow((l/s),mu2)
                  *(N(d1(n))-N(d2(n)))-pow((pow(l,(n+1.0))
                  /(pow(u,n)*s)),mu3(n))*(N(d3(n))-N(d4(n)));
         sum1+=pow((pow(u,n)/pow(l,n)),(mu1(n)-2.0))*pow((l/s),mu2)
                  *(N(d1(n)-sigma*sqrt(t))-N(d2(n)-sigma*sqrt(t)))
                  -pow(pow(l,(n+1.0))/(pow(u,n)*s),mu3(n)-2.0)
                  *(N(d3(n)-sigma*sqrt(t))-N(d4(n)-sigma*sqrt(t)));
      }
      c=s*exp((b-r)*t)*sum-x*exp(-r*t)*sum1;
         return c;
      public double uoDop(double stock, double strike, double up,
                        double low,double delt1, double delt2 )
      {
      s=stock;
      x=strike;
      delta1=delt1;
      delta2=delt2;
         u= up>s? (up*exp(sqrt(tau)*sigma*0.5826)):up;
         l= low<s? (low*exp(-sqrt(tau)*sigma*0.5826)):low;
         if(s>=u|s<=l)
         return 0.0;// no need to continue
         double mu2;
         int n=0;
      double sum=0.0;
      double sum1=0.0;
      double p;
      f=(l*exp(delta1*t));
      for(n=-5;n<6;n++)
         {
      mu2=2.0*n*((delta1-delta2)/(sigma*sigma));
      sum1+=pow((pow(u,n)/pow(l,n)),mu1(n))*pow((l/s),mu2)*(N(d2(n))
          -N(d1(n)))-pow((pow(l,(n+1.0))/(pow(u,n)*s)),mu3(n))
          *(N(d4(n))-N(d3(n)));
      sum+=pow((pow(u,n)/pow(l,n)),(mu1(n)-2.0))*pow((l/s),mu2)
          *(N(d2(n)-sigma*sqrt(t))-N(d1(n)-sigma*sqrt(t)))-pow
          (pow(l,(n+1.0))/(pow(u,n)*s),mu3(n)-2.0)*(N(d4(n)
          -sigma*sqrt(t))-N(d3(n)-sigma*sqrt(t)));
   }
           p=-s*exp((b-r)*t)*sum1+x*exp(-r*t)*sum;
           return p;
}
public double uiDinc(double stock, double strike, double up,
                    double low,double delt1, double delt2 )
{
   Blackscholecp b=new Blackscholecp(crate);
   b.bscholEprice(stock,strike,sigma,t,r);
   return (b.getCalle()-uoDoc(stock,strike,up,low,delt1,delt2));
}
public double uiDinp(double stock, double strike, double up,
                    double low,double delt1, double delt2 )
{
   Blackscholecp b=new Blackscholecp(crate);
   b.bscholEprice(stock,strike,sigma,t,r);
   double outvalue=uoDop(stock,strike,up,low,delt1,delt2);
   return b.getPute()-outvalue;
}
private double d1(double n)
{
   double dvalue=(log((s*pow(u,2.0*n))/(x*pow(l,2.0*n)))
                 +((b+((sigma*sigma)*0.5))*t))/(sigma*sqrt(t));
                 return dvalue;
}
private double d2(double n)
{
      double dvalue=(log((s*pow(u,2.0*n))/(f*pow(l,2.0*n)))
                  +((b+((sigma*sigma)*0.5))*t))/(sigma*sqrt(t));
                  return dvalue;
}
private double d3(double n)
{
```

```
double dvalue=(log((pow(l,2.0*n+2.0))/(x*s*pow(u,2.0*n)))
            +((b+((sigma*sigma)*0.5))*t))/(sigma*sqrt(t));
            return dvalue;
}
private double d4(double n)
{
   double dvalue=(log((pow(l,2.0*n+2.0))/(f*s*pow(u,2.0*n)))
               +((b+((sigma*sigma)*0.5))*t))/(sigma*sqrt(t));
               return dvalue;
}
private double mu1(double n)
{
   double mvalue= 2.0*(b-delta2-n*(delta1-delta2))/(sigma*sigma)+1.0;
   return mvalue;
}
   private double mu3(double n)
{
   double mvalue=2.0*(b-delta2+n*(delta1-delta2))/(sigma*sigma)+1.0;
   return mvalue;
}
private double N(double d)
{
   Probnorm p=new Probnorm();
      double probvalue=d>(6.95)?1.0:d<(-6.95)?0.0:p.ncDisfnc(d);
               //restrict the range of cdf values
   return probvalue;
}
```

Listing 15.1. Double Barrier Options

15.2. Valuing With a Single Put/Call Model

The Haug pricing formulae make use of single barrier pricing using a mechanism of put-call transformations. Following the observations of Bjerksund & Stensland (1993) for American options that:

$$C(S, X, T, r, b, \sigma) = P(X, S, T, r - b, -b, \sigma)$$

Where S is the asset price, X the strike price, b the cost of carry, r is the risk free rate and T the time to maturity. The call put transformation is therefore a call in terms of a put with the asset price being equal to the strike price and the strike price being substituted by the asset price, with the risk free rate replaced by $r - b$ and cost of carry $-b$. If we re-write the payoff from a call as

$\max(S - X; 0) \equiv X/S\max(S^2/X - S; 0)$ Then we can express the parity equation as:

$$C(S, X, T, r, b, \sigma) = X/S^{*}P(S, S^2/X, r - b, -b, \sigma)$$

Table 15.3. Put Up & In-Down & In valuations

S=110.0, X=110.0 r=0.05, b=0.05				Time 0.25				Time 0.5			
				σ1	σ2	σ3	σ4	σ1	σ2	σ3	σ4
U	L	δ1	δ2	0.10	0.15	0.25	0.35	0.10	0.15	0.25	0.35
150.0	60.0	0.0	0.0	0.000	0.000	0.0001	0.0287	0.000	0.000	0.0255	0.7828
140.0	70.0	0.0	0.0	0.000	0.000	0.0105	0.418	0.000	0.0004	0.3734	2.9587
130.0	80.0	0.0	0.0	0.000	0.0004	0.3049	2.2692	0.000	0.0464	2.0568	6.6114
120.0	90.0	0.0	0.0	0.0008	0.1277	2.3991	5.8693	0.0471	0.9673	5.3775	9.2216
110.0	100.0	0.0	0.0	1.5649	2.6321	4.7918	6.9584	1.8956	3.3639	6.3701	9.3956
U	L	−δ1	δ2								
150.0	60.0	0.05	0.05	0.000	0.000	0.0001	0.0409	0.000	0.000	0.0622	1.1095
140.0	70.0	0.05	0.05	0.000	0.000	0.0191	0.5305	0.000	0.0038	0.6768	3.7134
130.0	80.0	0.05	0.05	0.000	0.0017	0.4396	2.6186	0.0026	0.2067	2.9248	7.4285
120.0	90.0	0.05	0.05	0.0056	0.2676	2.8394	6.1623	0.3972	2.0002	5.9912	9.3507
110.0	100.0	0.05	0.05	1.5649	2.6321	4.7918	6.9584	1.8956	3.3639	6.3701	9.3956
U	L	δ1	−δ2								
150.0	60.0	0.05	0.05	0.000	0.000	0.000	0.0255	0.000	0.000	0.0241	0.7041
140.0	70.0	0.05	0.05	0.000	0.000	0.0094	0.3787	0.000	0.000	0.3406	2.7138
130.0	80.0	0.05	0.05	0.000	0.0004	0.274	2.1094	0.0001	0.0001	1.8764	6.2528
120.0	90.0	0.05	0.05	0.0008	0.1146	2.2198	5.6811	0.0548	0.0548	5.1141	9.1073
110.0	100.0	0.05	0.05	1.5649	2.6321	4.7918	6.9584	1.8956	1.8956	6.3701	9.3956

Table 15.4. Put Up & Out-Down & Out valuations

S=110.0, X=110.0 r=0.05, b=0.05				Time 0.25				Time 0.5			
				σ1	σ2	σ3	σ4	σ1	σ2	σ3	σ4
U	L	δ1	δ2	0.10	0.15	0.25	0.35	0.10	0.15	0.25	0.35
150.0	60.0	0.0	0.0	1.5649	2.6321	4.7917	6.9297	1.8956	3.3639	6.3446	8.6127
140.0	70.0	0.0	0.0	1.5649	2.6321	4.7813	6.5404	1.8956	3.3636	5.9967	6.4368
130.0	80.0	0.0	0.0	1.5649	2.6318	4.4869	4.6892	1.8955	3.3175	4.3133	2.7841
120.0	90.0	0.0	0.0	1.5641	2.5044	2.3927	1.0891	1.8485	2.3966	0.9926	0.174
110.0	100.0	0.0	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
U	L	−δ1	δ2								
150.0	60.0	0.05	0.05	1.5649	1.5649	4.7917	6.9174	1.8956	3.3639	6.3079	8.2861
140.0	70.0	0.05	0.05	1.5649	1.5649	4.7727	6.4279	1.8956	3.3601	5.6933	5.6822
130.0	80.0	0.05	0.05	1.5649	2.5649	4.3522	4.3398	1.893	3.1572	3.4453	1.9671
120.0	90.0	0.05	0.05	1.5592	1.5592	1.9524	0.796	1.4984	1.3637	0.3789	0.0449
110.0	100.0	0.05	0.05	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
U	L	δ1	−δ2								
150.0	60.0	0.05	0.05	1.5649	2.6321	4.7917	6.9328	1.8956	3.3639	6.3461	8.6915
140.0	70.0	0.05	0.05	1.5649	2.6321	4.7824	6.5797	1.8956	3.3634	6.0295	6.6817
130.0	80.0	0.05	0.05	1.5649	2.6317	4.5178	4.849	1.8955	3.3134	4.4937	3.1428
120.0	90.0	0.05	0.05	1.5641	2.5175	2.572	1.2773	1.8408	2.4726	1.256	0.2882
110.0	100.0	0.05	0.05	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

For a standard barrier 'in' option, the call put transformation can be written as:

$$\begin{aligned} C_{di}(S, X, H, r, b) &= P_{ui}(X, S, SX/H, r-b, -b) \\ &= X/S^{*} P_{ui}(S, S^2/X, S^2/H, r-b, -b) \end{aligned} \tag{15.2.1}$$

$$\begin{aligned} C_{ui}(S, X, H, r, b) &= P_{di}(X, S, SX/H, r-b, -b) \\ &= X/S^{*} P_{di}(S, S^2/X, S^2/H, r-b, -b) \end{aligned} \tag{15.2.2}$$

It can be argued that constructing a series of positions in up and in and down and in put options which protect the upper and lower barriers; we can value a double barrier call from a series of single barrier puts. The formula uses four such terms.

15.2.1. Valuing Double Calls

For the double barrier call:

$$\begin{aligned} &C_{ui,di}(S, X, L, U) \approx \min[C(S, X, T); C_{di}(S, X, L) + C_{ui}(S, X, U) \\ &- \frac{X}{U} P_{ui}\left(S, \frac{U^2}{X}, \frac{U^2}{L}\right) - \frac{X}{L} P_{di}\left(S, \frac{L^2}{X}, \frac{L^2}{U}\right) + \frac{U}{L} C_{di}\left(S, \frac{L^2 X}{U^2}, \frac{L^3}{U^2}\right) \\ &+ \frac{L}{U} C_{ui}\left(S, \frac{U^2 X}{L^2}, \frac{U^3}{L^2}\right) - \frac{LX}{U^2} P_{ui}\left(S, \frac{U^4}{L^2 X}, \frac{U^4}{L^3}\right) + \frac{UX}{L^2} P_{di}\left(S, \frac{L^4}{U^2 X}, \frac{L^4}{U^3}\right) \\ &+ \frac{U^3}{L^2} C_{di}\left(S, \frac{L^4 X}{U^4}, \frac{L^5}{U^4}\right) + \frac{L^2}{U^2} C_{ui}\left(S, \frac{U^4 X}{L^4}, \frac{U^5}{L^4}\right)] \end{aligned} \tag{15.2.3}$$

15.2.2. Valuing Double Put's

For the double barrier put:

$$\begin{aligned} &P_{ui,di}(S, X, L, U) \approx \min[P(S, X, T); P_{di}(S, X, L) + P_{ui}(S, X, U) \\ &- \frac{X}{U} C_{ui}\left(S, \frac{U^2}{X}, \frac{U^2}{L}\right) - \frac{X}{L} C_{di}\left(S, \frac{L^2}{X}, \frac{L^2}{U}\right) + \frac{U}{L} P_{di}\left(S, \frac{L^2 X}{U^2}, \frac{L^3}{U^2}\right) \\ &+ \frac{L}{U} P_{ui}\left(S, \frac{U^2 X}{L^2}, \frac{U^3}{L^2}\right) - \frac{LX}{U^2} C_{ui}\left(S, \frac{U^4}{L^2 X}, \frac{U^4}{L^3}\right) + \frac{UX}{L^2} C_{di}\left(S, \frac{L^4}{U^2 X}, \frac{L^4}{U^3}\right) \\ &+ \frac{U^3}{L^2} P_{di}\left(S, \frac{L^4 X}{U^4}, \frac{L^5}{U^4}\right) + \frac{L^2}{U^2} P_{ui}\left(S, \frac{U^4 X}{L^4}, \frac{U^5}{L^4}\right)] \end{aligned} \tag{15.2.4}$$

Table 15.5 shows example values using this approximation for a call up and-in down and-in option.

Table 15.5. Haug's approximation for a double barrier call option derived from a series of single barrier options

S=110.0, X=110.0 r=0.05, b=0.05				Time 0.25				Time 0.5			
				σ1	σ2	σ3	σ4	σ1	σ2	σ3	σ4
U	L	δ1	δ2	0.10	0.15	0.25	0.35	0.10	0.15	0.25	0.35
150.0	60.0	0.0	0.0	0.000	0.0024	0.5725	2.9716	0.000	0.230	3.4885	8.3165
140.0	70.0	0.0	0.0	0.000	0.0585	1.7359	4.9947	0.000	1.0313	5.6753	10.2166
130.0	80.0	0.0	0.0	0.0344	0.6906	3.8969	7.0688	0.7508	3.1248	7.8299	11.5704
120.0	90.0	0.0	0.0	1.2268	3.014	5.8356	8.222	3.4507	5.5872	9.0008	12.1115
110.0	100.0	0.0	0.0	2.9313	3.9986	6.1582	8.3248	4.6115	6.0798	9.086	12.1115

Table 15.6. Kunitomo & Ikeda direct calculation for a double barrier call option with flat boundaries

S=110.0, X=110.0 r=0.05, b=0.05				Time 0.25				Time 0.5			
				σ1	σ2	σ3	σ4	σ1	σ2	σ3	σ4
U	L	δ1	δ2	0.10	0.15	0.25	0.35	0.10	0.15	0.25	0.35
150.0	60.0	0.0	0.0	0.000	0.0024	0.5725	2.9716	0.000	0.230	3.4885	8.3165
140.0	70.0	0.0	0.0	0.000	0.0585	1.7359	4.9947	0.000	1.0313	5.6753	10.2166
130.0	80.0	0.0	0.0	0.0344	0.6906	3.8969	7.0688	0.7508	3.1248	7.8299	11.5704
120.0	90.0	0.0	0.0	1.2268	3.014	5.8356	8.222	3.4507	5.5872	9.0008	12.1115
110.0	100.0	0.0	0.0	2.9313	3.9986	6.1582	8.3248	4.6115	6.0798	9.086	12.1115

Table 15.7. Differences between Haug's approximation and Kunitomo & Ikeda formula for option valuation (Haug-Kunitomo & Ikeda)

Time = 0.25				Time = 0.5			
σ1	σ2	σ3	σ4	σ1	σ2	σ3	σ4
0.10	0.15	0.25	0.35	0.10	0.15	0.25	0.35
−0.0000001	−0.0000285	0.000000	0.000000	−0.001748	−0.000025	0.000000	0.000000
−0.0001215	−0.0000007	0.000000	0.000000	−0.054508	0.000000	0.000000	0.0000472
−0.0001492	0.0000001	0.000000	0.0000285	0.0048285	0.000000	0.0000619	0.0081228
−0.0000012	0.000000	0.0003737	0.0136462	0.0000001	0.0000347	0.0273155	0.0183606
0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Table 15.6 shows the valuations for direct calculation and Table 15.7 shows comparative differences.

For lower volatilities the errors show that the Haug approximation is lower. Whereas at higher volatilities, the approximation is higher

Haug's double barrier approximation methods are shown in Listing 15.2.

```
package FinApps;
import static java.lang.Math.*;
import java.text.*;
import java.io.*;

public class Dbarrierh {

   public Dbarrierh(double rate,double q,double time,
                     double volatility, int period ) {
      crate=q;
      r=rate;
      t=time;
      //0 continuos,1 hourly,2 daily,3 weekly, 4 monthly
      sigma=volatility;
      tau=period==1?(1.0/(24*365)):period==2?(1.0/(365.0)):
      period==3?(1.0/(52.0)):period==4?(1.0/(12.0)):0.0;
      b=crate==0.0?0.0:(b=crate!=r?(r-crate):r);
}
double b;
double tau;
double crate;
double t;
double r;
double s;
double u;
double x;
double l;
double sigma;
double f;
double delta1;
double delta2;
double xu1;
double xu2;
double xu3;
double xu4;
double up1;
double up2;
double up3;
double up4;
double xl1;
double xl2;
double xl3;
double xl4;
double low1;
double low2;
double low3;
double low4;

public double uiDinc(double stock, double strike, double up, double low ) {
   s=stock;
   x=strike;
   paraMs(up,low,x);
   Blackscholecp b=new Blackscholecp(crate);
   b.bscholEprice(stock,strike,sigma,t,r);
   double bscall=b.getCalle();
```

```
    Sbarrierop sb=new Sbarrierop(r,crate,t,0);
    double cui=sb.uplncall(s,x,sigma,up,0.0);
    double cdi=sb.downlcall(s,x,sigma,low,0.0);
    double pui1=sb.uplput(s,xu1,sigma,up1,0.0);
    double pdi1=sb.downlput(s,xl1,sigma,low1,0.0);
    double cdi1=sb.downlcall(s,xl2,sigma,low2,0.0);
    double cui1=sb.uplncall(s,xu2,sigma,up2,0.0);
    double pui2=sb.upIput(s,xu3,sigma,up3,0.0);
    double pdi2=sb.downIput(s,xl3,sigma,low3,0.0);
    double cdi2=sb.downIcall(s,xl4,sigma,low4,0.0);
    double cui2=sb.upIncall(s,xu4,sigma,up4,0.0);
    double term=(cui+cdi-(x/up)*pui1-(x/low)*pdi1+(up/low)*cdi1
                +(low/up)*cui1-(low*x/(up*up))*pui2-(up*x/(low*low))
                *pdi2+(up*up/(low*low))*cdi2+(low*low/(up*up))*cui2);
    return(min(bscall,term));
}
public double uiDinp(double stock, double strike, double up, double low ){
    s=stock;
    x=strike;
    paraMs(up,low,x);
    Blackscholecp b=new Blackscholecp(crate);
    b.bscholEprice(stock,strike,sigma,t,r);
    double bscall=b.getPute();
    Sbarrierop sb=new Sbarrierop(r,crate,t,0);
    double pui=sb.uplput(s,x,sigma,up,0.0);
    double pdi=sb.downlput(s,x,sigma,low,0.0);
    double cui1=sb.uplncall(s,xu1,sigma,up1,0.0);
    double cdi1=sb.downlcall(s,xl1,sigma,low1,0.0);
    double pdi1=sb.downlput(s,xl2,sigma,low2,0.0);
    double pui1=sb.uplput(s,xu2,sigma,up2,0.0);
    double cui2=sb.uplncall(s,xu3,sigma,up3,0.0);
    double cdi2=sb.downlcall(s,xl3,sigma,low3,0.0);
    double pdi2=sb.downlput(s,xl4,sigma,low4,0.0);
    double pui2=sb.uplput(s,xu4,sigma,up4,0.0);
    double term=(pui+pdi-(x/up)*pui1-(x/low)*pdi1+(up/low)*cdi1+(low/up)
                *cui1-(low*x/(up*up))*pui2-(up*x/(low*low))
                *pdi2+(up*up/(low*low))*cdi2+(low*low/(up*up))*cui2);
    return(min(bscall,term));
}
public double uoDoc(double stock, double strike, double up, double low ) {
    if(stock>=up|stock <=low)
        return 0.0;// no need to continue
    Blackscholecp b=new Blackscholecp(crate);
    b.bscholEprice(stock,strike,sigma,t,r);
    double bscall=b.getCalle();
    double val1=(bscall-uiDinc(stock,strike,up,low));
    return val1;
}
public double uoDop(double stock, double strike, double up, double low ) {
    if(stock>=up|stock<=low)
        return 0.0;// no need to continue
    Blackscholecp b=new Blackscholecp(crate);
    b.bscholEprice(stock,strike,sigma,t,r);
    double bscall=b.getPute();
```

```
   double val1=(bscall-uiDinp(stock,strike,up,low));
   System.out.println(" r=="+r+" b=="+crate);
   return val1;
}
private void paraMs(double up, double low, double x) {
   xu1=((up*up)/x);
   xu2=((up*up*x)/(low*low));
   xu3=(pow(up,4)/(low*low*x));
   xu4=((pow(up,4)*x)/(pow(low,4)));
   up1=((up*up)/low);
   up2=(pow(up,3)/(low*low));
   up3=(pow(up,4)/(pow(low,3)));
   up4=(pow(up,5)/(pow(low,4)));
   xl1=((low*low)/x);
   xl2=((low*low*x)/(up*up));
   xl3=(pow(low,4)/(up*up*x));
   xl4=((pow(low,4)*x)/(pow(up,4)));
   low1= ((low*low)/up);
   low2=(pow(low,3)/(up*up));
   low3=(pow(low,4)/(pow(up,3)));
   low4=(pow(low,5)/(pow(up,4)));
}
```

LISTING 15.2. Double Barrier Approximation

References

Bjerksund and Stensland (1993). "American Exchange Options and a put call Transformation: A Note," *Journal of Business Finance and Accountancy*, 20(5), 61–64.

Geman, H. and M. Yor (1996). "Pricing and Hedging Double-Barrier Options: A Probabilistic Approach," *Mathematical Finance*, 6(4), 365–378.

Haug, E. H. (1999). "Barrier Put-Call Transformations," *Tempus Financial Engineering*.

Kunitomo, N. and M. Ikeda (1992). "Pricing Options with Curved Boundaries," *Mathematical Finance*, 2(4), 275–2.