

Implicit Finite Differences Method For Pricing Barrier Option

Fabien Le Floc'h - fabien @ 31416.org

Why?

While trying to price a simple knock down and out barrier option, I encountered several difficulties I did not expect with the implicit finite differences method. The explicit method has less issues with barrier options pricing. I will show here what the tricky parts are and why explicit seems simpler in this case.

I am not the only one to have encountered unexpected problems with the implicit method and barrier options, you can find many posts about it on Wilmott forums without the correct answer.

The problem

Pricing vanilla option under implicit finite difference is not very difficult. It is well presented in many books (including Hull). The only tricky bit is the boundary conditions. But for a vanilla call/put they are trivial.

Wilmott in his book on quantitative finance presents alternative boundary conditions that can be used with more exotic options. One is the null gamma when the asset price is high, another is the rate drift when the asset price is 0 (less useful).

We will consider here a Knock Down and Out Call Barrier Option.

For such an option, the payoff is the rebate R if we are under the barrier level L. It seems natural to just take the vanilla implicit finite difference code and impose price=R when spot<=L.

Unfortunately this won't give the right price. The resulting price won't even be stable. If you have successfully priced a barrier option with the explicit method, you know that your grid has to hit the barrier level L to give a good and stable price. But **this is not enough for the implicit method**.

Truncating the grid at the barrier level will give a good and stable price. This is the important bit. Now let's find out what's different.

Implicit method on barrier option - not truncated						
	M-2	M-1	maturity M	M from M-1 equa	pd	-2.3670
SI-2	0	0	0	0	pm	5.75504
SI-1	0	0	0	0	pu	-2.3841
barrier SI	0	0	0	-16.2370308853		
Sn-1	6.56375034755527	6.81069184743174	7.46794344808887	16.00586670042		
initial spot (Sn)	9.53654775371325	9.72712246736757	9.9999999999992	9.99999999999919		
Sn+1	12.4241369017133	12.5245456354425	12.5978352085146	12.5978352085146		
	15.269792839259	15.2922244790516	15.263157894736	15.263157894736		
	18.1112979286282	18.0779492153183	17.9977212721206	17.9977212721206		

Implicit method on barrier option - truncated						
	M-2	M-1	maturity M	M from M-1 equa	pd	-2.3670
barrier SI	0	0	0	0	pm	5.75504
Sn-1	4.15181658714879	4.913980688917	7.467943448089	7.467943448089	pu	-2.3841
initial spot (Sn)	7.96119839790346	8.72977421955	9.9999999999992	9.9999999999992		
Sn+1	11.4342977022039	12.0001096806266	12.5978352085146	12.5978352085146		
	14.6643981096676	15.0164601496857	15.263157894736	15.263157894736		
	17.7483162079836	17.932943967962	17.9977212721206	17.9977212721206		
	20.762503258783	20.8314763492853	20.8033240997221	20.8033240997221		
	23.7610273122554	23.7560286475001	23.6818118653901	23.6818118653901		

The variables pd, pm, pu are defined such as:

$$pd*price(M-1,n-1)+pm*price(M-1,n)+pu*price(M-1,n+1) = price(M, n).$$

We use the log of asset prices, this makes pd,pm,pu constants. The reasoning would be the same in linear asset price scale. This equation is used in M from M-1 equations column to display what the initial (unsolved) equation gives.

Obviously the problem is at the boundary conditions. The delta is not defined at the barrier, this is what creates the instability as the finite differences algorithm assumes the delta is defined everywhere. You can see in bold red the linear equations are not solved properly at the boundary. This is why truncating gives the correct price. The truncated version does not make use of a delta approximation at the barrier level as the equation is not solved there, the boundary condition only is used in the truncated algorithm.

This apparently localized effect actually propagates quite far as you can see with the difference of option prices at time M-2.

Why explicit method has not this problem.

Explicit method on barrier option - not truncated						
	M-2	M-1	maturity M	M-1 from M equa	pd	
SI-2	0	0	0	0	pm	0.47340
SI-1	0	0	0	0	pu	0.47681
barrier SI	0	0	0	3.56079033181299		
Sn-1	5.03488124938312	5.133967490419	7.46794344808922	5.133967490419		
initial spot (Sn)	8.94343409906935	10.0320026721801	9.99999999999956	10.0320026721801		
Sn+1	12.6597212370074	12.628798816031	12.597835208515	12.628798816031		
	15.3229122339189	15.2930554444001	15.2631578947364	15.2930554444001		
	18.0552885442183	18.0265250695062	17.997721272121	18.0265250695062		

Here pd,pm,pu are different because they follow the explicit equation:

$$pd*price(M,n-1)+pm*price(M,n)+pu*price(M,n+1) = price(M-1, n).$$

The equation is already solved for us. The bad point in red is overridden in the next run by the barrier rebate (=0). So it will have no impact.

The fictitious point

There is a way to have implicit method working with a non truncated grid, it is by ensuring the price is correct at the barrier by including some fictitious points.

If we just take table 1 and change the row under the barrier level so that we have the the boundary condition still works linearly (i.e we assume a delta and make it so that price @ L = 0), it means we have:

$$price(M-1,L-1)=-pu/pd*price(M-1,L+1).$$

Replacing in our first table gives:

Implicit method on barrier option - fictitious point added						
	M-2	M-1	maturity M	M from M-1 equa	pd	
SI-2	0	0	0	0	pm	-2.3670
SI-1	0	-6.85977608094	0	-39.47829622992	pu	5.75504
barrier SI	0	0	0	3.55271368E-15		-2.3841
Sn-1	6.56375034755527	6.81069184743174	7.46794344808887	16.00586670042		
initial spot (Sn)	9.53654775371325	9.72712246736757	9.9999999999992	9.99999999999919		
Sn+1	12.4241369017133	12.5245456354425	12.5978352085146	12.5978352085146		
	15.269792839259	15.2922244790516	15.263157894736	15.263157894736		
	18.1112979286282	18.0779492153183	17.9977212721206	17.9977212721206		

It is almost correct but there is still a problem at L+1. This is because we replaced in the table, we actually need to solve the equation again. Solving the equation will ensure price(L+1) is correct.

Implicit method on barrier option - fictitious point added						
	M-2	M-1	maturity M	M from M-1 equa	pd	-2.3670
SI-2	0	0	0	0	pm	5.75504
SI-1	0	-4.94939544281	0	-28.48397632002	pu	-2.3841
barrier SI	0	0	0	1.776356839E-15		
Sn-1	6.56375034755527	4.913980688917	7.46794344808887	7.467943448089		
initial spot (Sn)	9.53654775371325	8.72977421955	9.9999999999992	9.9999999999992		
Sn+1	12.4241369017133	12.0001096806266	12.5978352085146	12.5978352085146		
	15.269792839259	15.0164601496857	15.263157894736	15.263157894736		
	18.1112979286282	17.932943967962	17.9977212721206	17.8159418434154		

We have the same prices as the truncated version.

Of course the price under the barrier will be wrong, because of the fictitious point.

The exact same thing will happen to Crank Nicolson as Crank Nicolson is just a mix of explicit and implicit methods.

Annexe – Parameters used

```

dividendRate = 0.04;
discountRate = 0.08;
vol = 0.25;
expiryTime = 0.5;
isAmerican = false;
initialSpot = 100;
isCall = true;
strike = 90;
level = 95;
dx = Math.log(initialSpot/level)/2; //very important S0*M*dx hit barrier
timeSteps = 10;

```