Source | Model | Option
| Model_Option | Help on tr methods | Archived Tests

# tr_dermankani

Input parameters:

- StepNumber $N$

Output parameters:

- Price
- Delta

This algorithm is taken from [1]. It gives a solution to the problem of pricing Barrier Options with a Tree method. The issue, as it is discussed there, is to provide the tree algorithm with a better value of the barrier than the crude approximation with the nearest level in the tree: indeed the spacing of the tree grid in the space direction is of order $\sqrt{h}$ so that the naive barrier-version of the CRR algorithm, for instance, can not yield a convergence order better than $\sqrt{h}$. Numerical tests ([1]) show that this is definitely too crude. The idea of the algorithm as discussed there is to take as a price value for the node just above the barrier (let us denote this level $S_0 d^{eta0}$) the linear interpolation between the price in case the barrier is exatcly at the level of the node ("modified barrier" $mod = S_0 d^{eta0}$-the price of the option is then given by the *rebate* by definition of the contract) and the price in case the barrier is at the level of the node below ("effective barrier" $eff = S_0 d^{eta0+1}$):

$$C = \frac{(L - \text{eff})}{(\text{mod} - \text{eff})} C\,(L = \text{mod}) + \frac{(\text{mod} - L)}{(\text{mod} - \text{eff})} C\,(L = \text{eff})$$

This is a CRR tree with a modification of the backward expectation formula at the critical mesh breaching the barrier.

/*Price, intrinsic value arrays*/
This is a flat tree so we store the intrinsic values in an array `iv` to avoid recomputation at each node.

/*Up and Down factors*/
Exactly those of CRR.

/*Risk-Neutral Probability*/
Same remark.

/*Number of down moves just before breaching the barrier*/
Here we compute the value of the integer $eta0$ above. Note that it may be 0, ie the barrier is breached at the first mesh of the tree. This requires a particular handling of the calculation of the delta.

/*Weights for the linear interpolation at the critical node*
/*Node above the barrier*/
Here we compute the above weights $pl = \frac{(L-\mathrm{eff})}{(\mathrm{mod}-\mathrm{eff})}$

/*Intrinsic value initialization*/
We start the indexing from above. The index of the critical node is `N+eta0` . There is no need to compute the intrinsic values for the nodes below.

/*Terminal Values*/
`npoints` is the index of the critical node among the values of the underlying at maturity. We store the value of the option at the effective barrier $eff$ in `P[npoints+1]` .

/*Backward Resolution*/
We begin with the case `eta0>0` where the first mesh does not breach the barrier. In this case there are 3 stages: first the barrier is active, then (backward) the barrier is strictly below the lower node of the tree and the algorithm is exactly that of a CRR tree.

/*First part-the barrier is active*/
Every two steps in time the lower mesh of the tree (with root node index `npoints` breaches the barrier (`odd=1`) or not (`odd=0`). In the breaching case, the computation

```
P[j]=pu*P[j]+ pd*P[j+1];
if (am)
 P[j]=MAX(iv[i+2*j],P[j]);
```

for $j = npoints$ computes the value $C\,(L = \mathrm{eff})$. Notice that we use here the value P[npoints+1] computed before. Then we implement the interpolation in : $P[npoints] = pl * rebate + one_minus_pl * P[npoints]$ Next if necessary we store the value of the $rebate$ in P[npoints+1]

/*Second part-the barrier is strictly below the tree*/
From now on this is the standard CRR algorithm (cf. Routine tr_coxrossrubinstein_c).

/*Delta*/

/*First time step*/

/*eta0=0, the first mesh breaches the barrier*/
/*The barrier is always active*/
Then in order to compute the delta we stop the above First part algorithm at the first time step. We store the price value at the node $(2h, S0)$ in `flat`price.

/*For the delta*/
Here we store the value at the node $(h, u * S0)$ in `up`price.

/*First Time Step*/
Since the first mesh breaches the barrier we use the interpolation formula.

/*Price*/

/*Delta*/
The delta is computed as a finite-difference approximation, at time $h$, between the value of the price at node $(h, u * S0)$ and at node $(h, S0)$. The value of the price at this last node is the interpolation between the price at the nodes $(0, S0)$ and the nodes $(2h, S0)$.

/*Memory Desallocation*/

# References

[1] E.DERMAN I.KANI D.ERGENER I.BARDHAN. Enhanced numerical methods for options with barriers. *Financial Analyst Journal*, pages 65–74, Nov-Dec 95 1995. 1