ADD GREEKS TO BINOMIAL TREE ENGINES

Ismail EL HADRAMI, Sami RMILI

E-mail: ismail.el-hadrami@imt-atlantique.net,sami.rmili@imt-atlantique.net

1 Introduction

Greeks are quantities that represent the sensitivity of the price of derivative securities with respect to changes in the price of underlying assets or parameters. They are defined by derivatives of the option price function with respect to parameters such as the price of underlying assets, volatility level, and spot interest rate.

To briefly explain the binomial tree model, it is a computational method for pricing options on securities whose price process is governed by the geometric Brownian motion.

2 Main Issue

The main problem of binomial tree model is that as currently implemented, there's a single point at $t = 0$ as shown in figure 1(a). This makes it impossible to calculate quickly and accurately the Delta and Gamma of an option; we can either re-run the calculation with a perturbed value of the underlying asset (which is slow, since we need three full calculations for Delta and Gamma) or we can use points at the first time step after $t = 0$ to obtain option values for a perturbed underlying which is not accurate, since the option value at those points would be taken at $t \neq 0$.

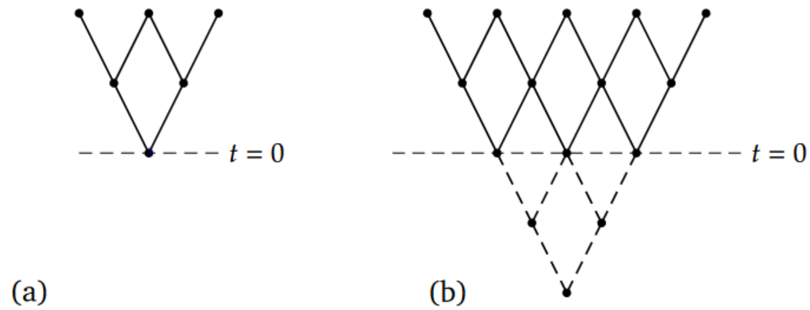

Figure 1: trees with one (a) and three (b) points at t = 0.

3 Solution

The problem can be solved by having three points for $t = 0$ with the center point at u_0, as shown in figure 1(b), so that we can obtain the option values for $u + \delta u$ and $u - \delta u$ to use in the formulas for Delta and Gamma.

3.1 Method

We begin by changing the *size()* function in the *BinomialTree* class as such :

```
Size size(Size i) const {
            return i+3;
}
```

We also need to change the index of calculation of the underlying price since every layer is now of size 3. The idea is to shift the said index by *-1*.

```
Size shifted_index = index - 1;
BigInteger j = 2*BigInteger(shifted_index) - BigInteger(i);
```

Listing 1: Index shift

The modification is done for all the binomial tree base classes.

Last but not least, we need to update how the delta and gamma are calculated according to the change :

```
option.rollback(0.0);
Array va(option.values());
QL_ENSURE(va.size() == 3, "Expect 3 nodes in grid at time t=0");
Real p0u = va[2]; // up
Real p0m = va[1]; // mid
Real p0d = va[0]; // down (low)
Real s0u = lattice->underlying(0, 2); // up price
```

```
Real s0m = lattice->underlying(0, 1); // middle price
Real s0d = lattice->underlying(0, 0); // down (low) price

// Calculate delta
Real delta = (p0u - p0d) / (s0u - s0d);

// Calculate gamma
Real delta0u = (p0u - p0m)/(s0u - s0m);
Real delta0d = (p0m - p0d)/(s0m - s0d);
Real gamma = (delta0u - delta0d) / ((s0u-s0d)/2);



// Store results
results_.value = p0m;
results_.delta = delta;
results_.gamma = gamma;
results_.theta = blackScholesTheta(process_,
                                    results_.value,
                                    results_.delta,
                                    results_.gamma);
```

Listing 2: Calculating delta then gamma

3.2 Results

In this section, we evaluate the performance of the new calculation method of delta and gamma. To do so, we'll compare the **NPV**, **Delta**, **Gamma** and the execution time. We do this for different Binomial tree base classes.

We begin by presenting these values for the initial method of calculation :

Tree type	**NPV**	Δ	Γ	**Exec. Time** (μs)
JarrowRudd	4.20500	-0.849518	0.0720572	251
CoxRossRubinstein	4.20472	-0.849587	0.0720719	174
AddEQPBinTree	4.20559	-0.849272	0.0720707	166
Trigeorgis	4.20473	-0.849586	0.0720719	163
Tian	4.20488	-0.849297	0.0721481	331
LeisenReimer	4.19732	-0.850354	0.0727937	332
Joshi4	4.19732	-0.850355	0.0727937	414

Table 1: Initial method results

After applying the modifications to the code, we get the following results :

Tree type	**NPV**	Δ	Γ	**Exec. Time** (μs)
JarrowRudd	4.20500	-0.846675	0.0721635	249
CoxRossRubinstein	4.20472	-0.84679	0.0721567	141
AddEQPBinTree	4.20559	-0.846429	0.0721755	142
Trigeorgis	4.20473	-0.846789	0.0721567	142
Tian	4.20488	-0.846834	0.0720787	340
LeisenReimer	4.19732	-0.850244	0.0716231	341
Joshi4	4.19732	-0.850244	0.0716231	340

Table 2: Results after modifications

4 Conclusion

The **NPV** values for the two methods are identical. However, the second method provides an interesting compromise between precision and time of calculation. In fact, it returns approximate values of *gamma* and *delta*, but with less computing time than the first method.