**Black Scholes PDE**

We have delta hedged portfolio (P) of 2 positions:

* short position in one Option (V)
* and long position in shares

We can notice that term has cancelled which means that stochastic term is removed (uncertainty is removed) only deterministic terms remains. Therefore, the portfolio must yield return to other riskless instruments, otherwise there will be an arbitrage opportunity. Therefore:

We have Black Scholes PDE:

**Finite Difference Scheme**

**Forward Difference**

* For time derivative
* For first spot derivative use central difference
* For second spot derivative use

Moreover

* dS = ∆S
* dt = ∆t
* M∆S = S\_max
* N∆t = T
* V(i,j) denotes V(i∆t, j∆S)

BS PDE

Note, that this becomes the Backward Euler using the Forward Difference which can be confusing, to say the least! This is also called Implicit Euler, since the formula for does not have explicit solution, i.e. need to use matrix algebra to solve at each time step. The Implicit method is more stable!

**Boundary Conditions**

V(n,j) = max(j∆S – K ,0)

V(i,0) = 0

V(i,M) = S\_max - Ke^(-r\*(i∆t))

**Solution**

To solve, you have to start solving for unknowns backward in time (starting from maturity time T), since the boundary conditions are known for V(n,j) = max(j∆S – K ,0). In that case you have three unknowns and one knows variable give the equation:

So we need to use Matrix Algebra to solve for the unknown values.

We can rewrite in matrix form:

We can further write as:

Then we can solve:

**Backward Difference**

* For first spot derivative use central difference
* For second spot derivative use

Moreover

* dS = ∆S
* dt = ∆t
* M∆S = S\_max
* N∆t = T
* V(i,j) denotes V(i∆t, j∆S)

BS PDE

Note, that this becomes the Forward Euler using the Backward Difference which can be confusing, to say the least! This is also called Explicit Euler, since this is an explicit formula for . The Explicit method can be very unstable! Be Careful! If you make dS step small, you have to make dt step very small! But you have to make dS step relatively small, otherwise you will have poor accuracy, but this makes the computation very expensive.

**Boundary Conditions**

V(n,j) = max(j∆S – K ,0)

V(i,0) = 0

V(i,M) = S\_max - Ke^(-r\*(i∆t))

**Solution**

Here we can solve with simple loop as the unknown variable at time i-1 is a result of known variables at time I, and since we start solving backwards in time (since the boundary condition is knows as T goes to maturity), we can start solving with a simple loop, no need for matrix notation.

**IMPORTANT!!!**

We have stability problem with Backward Euler in this case if becomes less than 0.5. This is similar to the properties of Forward Euler in Heat Equation PDE.

**Crank Nicolson**

Here we will take weighted average of forward and backward difference.

Forward difference:

Lets deduct td :

Backward Difference:

So now we take weighted average of both:

T from one side, t-dt from the other side:

So…

So…

So…

We solve via matrix

Then we can solve:

Or to be more easier for programming: