CREATIVE MATH.
13 (2004), 31 - 36

Numerical methods for solving a Black-Scholes equation

Ioana Chiorean

Abstract. The Black-Scholes equation is one of the most used formula for evaluating European call options without dividents. The aim of this paper is to study the possibility of a parallel execution in order to obtain a numerical approximation of the solution.

1. Introduction

It is known that the Black-Scholes equation is used to determine the value of an option. It provides quantitative information to continuously buy or sell assets to maintain a portfolio that grows at the riskless rate and thus provides insurance against down turns in the value of assets held along or protect against a risk in the value of assets held short. Under this circumstances, the portfolio is hedged against looses and so the option serves as an insurance policy.

According with [2], [4], [5], [6], the Black-Scholes equation for evaluating the value of an option is:

$$\frac{\partial V}{\partial t}+\frac{1}{2}\sigma^2 S^2\frac{\partial^2 V}{\partial S^2}+rS\frac{\partial V}{\partial S}-rV=0 \tag{1.1}$$

where V denotes the value of an option and it is a function of the current value of the underlying asset, S, and time, t: $V=V(S,t)$; parameter σ indicates the volatility of the underlying asset and r is the interest rate.

In order to have an unique solution, which is the desired value of the option, some boundary conditions must be imposed, e.g.

$$V(S,t)=V_a(t) \text{ on } S=a \text{ and } V(S,t)=V_b(t) \text{ on } S=b \tag{1.2}$$

where V_a and V_b are given functions of t.

Also, there exist a final condition in time, such as:

$$V(S,t)=V_T(S) \text{ on } t=T \tag{1.3}$$

where T is the expiry and V_T is a known function.

Note 1. If the equation is "backward" in time, we impose a final condition like (1.3). If it is "forward" in time, we impose an "initial" condition on $t=0$.

2. The Black-Scholes equation for an European call option

The simplest financial option, an European call option, is a contract with the following conditions [6]:

- At a prescribed time in the future, known as the **expiry date**, the holder of an option may:
  - purchase a prescribed asset, known as the **underlying asset**, for a
  - prescribed amount, known as the **exercise price**.

Note 2.1. The option **to buy** an asset is known as a **call** option. Then exist, also, the option to **sell** an asset, known as a **put** option.

Note 2.2. There are, also, other call and put options, not only the European ones, e.g. the American options. It has nothing to do with the continents! An American option is one that may exercised at **any time** prior to expiry. An European option may be exercised only **at** expiry.

In what follows, we take into account an European call option, with the value $C(S,t)$.

3. The finite-difference method

The Black-Scholes equation can be solved analytically for some options, but for others, a numerical attempt is preferred. The most common method used is that of the finite-difference. It constitutes a very powerful and flexible technique and, if applied correctly, generates accurate numerical solutions.

In order to get the discretized form of (2.1), as in [6], we use the mesh, like in Figure 1. The nodes along the S-axis are spaced by distance ΔS and along the t-axis are spaced by Δt. The plane (S, t) will be divided in the **mesh points**, of the form $(n\Delta S, m\Delta t)$. Then we denote

$$C_n^m = C(n\Delta S, m\Delta t)$$

for the value of $C(S, t)$ at the mesh point $(n\Delta S, m\Delta t)$.

Figure 1. The mesh for the finite difference approximation.

Under this circumstances, using a forward difference in time and a centered one in space (for S), we get the following discretized Black-Scholes equation:

$$\frac{C_n^{m+1} - C_n^m}{\Delta t} + \frac{1}{2}\sigma^2 S^2 \frac{C_{n+1}^m - 2C_n^m + C_{n-1}^m}{(\Delta S)^2} + rS\frac{C_{n+1}^m - C_n^m}{\Delta S} - rC_n^m = 0 \tag{3.1}$$

with the boundary conditions:

$$\begin{aligned} C_n^m &= C_{N^-}^m = C_{-\infty}(N^-\Delta S, m\Delta t),\ 0 < m \le M \\ C_n^m &= C_{N^+}^m = C_{\infty}(N^+\Delta S, m\Delta t),\ 0 < m \le M \\ C_n^0 &= C_0(n\Delta S),\ N^- \le n \le N^+ \end{aligned} \tag{3.2}$$

Note 3.1. To solve the problem, we get a finite but large enough, number of S-steps:

$$N^-\Delta S \le S \le N^+\Delta S$$

where N^- and N^+ are large positive integers.

Note 3.2. The time axis will be divided, to expiry, into M equal time-steps, so that

$$\Delta t = \frac{1}{2}\sigma^2 \cdot T/M.$$

Making the calculus in (3.1), we get the discretized equation:

$$C_n^{m+1} = AC_{n+1}^m + BC_n^m + DC_{n-1}^m \tag{3.3}$$

where

$$\begin{aligned}
A &= -\frac{S\cdot\Delta t}{(\Delta S)^2}\cdot\left(\frac{1}{2}\sigma^2\cdot S + r\cdot\Delta S\right)\\
B &= ((\Delta S)^2+\sigma^2\cdot S^2\cdot\Delta t + r\cdot S\cdot\Delta t\cdot\Delta S + r\cdot\Delta t(\Delta S)^2)/(\Delta S)^2\\
D &= -(\sigma^2\cdot S^2\cdot\Delta t)/2(\Delta S)^2
\end{aligned}\tag{3.4}$$

with the boundary conditions given by (3.2).

One can see that (3.3) is obtained by an explicit finite difference discretization, it means that, if at time-step m we know C_n^m for all values n, we can explicitly calculate C_n^{m+1}, because it depends only on C_{n+1}^m, C_n^m and C_{n-1}^m, as shown in Figure 2.

Figure 2. Explicit finite-difference discretization

4. Algorithmic aspects of computing the numerical approximation for the Black-Scholes equation

A simple pseudo-code, using one processor, can be written for the explicit finite difference equation (3.3):

```
{Compute A, B, D according with (3.4)}
for n:=Nminus to Nplus do
    oldC[n]:=C_zero(n*dS);
for m:=1 to M do
begin
    newC[Nminus]:=C_m(Nminus*dS,m*dt);
    newC[Nplus]:=C_p(Nplus*dS,m*dt);
    for n:=Nminus to Nplus do
        newC[n]=A*oldC[n+1]+B*oldC[n]+D*oldC[n-1];
    for n:=Nminus to Nplus do
    oldC[n]:=newC[n];
end
```

It is clear that the execution time is of order $O(M*(Nplus-Nminus))$. It can be reduced by means of parallel calculus, using several processors (see [1]). In [3], a parallel approach is given, based on the binary method. Because every C_n^{m+1} depends upon three values of the previous time-step m, our idea is to use a linear connectivity among processors, like in Figure 3.

Figure 3. The linear network

According with this type of network, every processor P_n communicates with its neighbours from the left and right, P_{n-1} and P_{n+1}. If we think that every processor memorizes the precomputed values, A, B and D according with (3.4), and the boundary conditions (3.2), then, using $p = Nplus - Nminus + 1$ processors, we compute (3.3) in the following manner:

```
for i:=1 to p in parallel do
    oldC[i]:=C_n_zero(n*dS)
for m:=1 to M do
    for i:=1 to p in parallel do
    begin
        newC[n]:=A*oldC[n+1]+B*oldC[n]+D*oldC[n-1];
        oldC[n]:=newC[n]
    end;
```

It is clear that the time of execution is of order M, so we get a linear speed-up.

References

[1] Chiorean, I., *Calcul paralel*, Ed. Microinformatica, Cluj, 1994

[2] Etheridge, A., *A Course in Financial Calculus*, University Oxford

[3] Gerbessiotis, A., *Parallel option price valuation with the explicit finite difference method*, Dep. of Comp. Science, New Jersey Inst. of Technology, Research Report CS-02-04

[4] Manole, S., *Determinarea valorii unei opţiuni de tip European*, Univ. C-tin Brâncoveanu, Piteşti

[5] Watkins, T., *Derivation of the Black-Scholes equation for option value*, San Jose State University

[6] Wilmott, P., et al., *The Mathematics of Financial Derivatives*, Cambridge Univ. Press, 1995

Babeş-Bolyai University
Faculty of Mathematics
Kogălniceanu 1, 400084 Cluj-Napoca, Romania
E-mail address: ioana@cs.ubbcluj.ro