Applied Computational Finance

Lesson 2 - Finite Difference Methods in Option Pricing

Giovanni Della Lunga

Laurea Magistrale in Quantitative Finance

Bologna - April-May 2021

Table of Contents

Objectives

- In this lesson we give a gentle introduction to partial differential equation (PDEs);
- It can be considered to be a panoramic view and is meant to introduce some notation and example;
- We'll see also some example of Ordinary Differential Equations (ODEs) which can be considered a particular case of a PDE;
- In particular we examine PDEs in one or more space dimensions and a single time dimension;
- An example of a PDE with a derivative in time dimension is the heat equation in two spatial dimensions:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \tag{1}$$

Objectives

- We classify PDEs into three categoris of equation, namely *parabolic*,*hyperbolic* and *elliptic*;
- Parabolic equations are important for financial engineering applications because the Black-Scholes equation is a specific instance of such a category;
- A partial differential equation of second order—second-order, linear, constant-coefficient PDE for u takes the form

$$A u_{xx} + 2B u_{xy} + C u_{yy} + D u_x + E u_y + F = 0 \tag{2}$$

this PDE is classified as being **parabolic** if the coefficients satisfy the condition

$$B^2 - AC = 0$$

Derivation of the Black–Scholes PDE

- The following derivation is given in Hull's Options, Futures, and Other Derivatives that, in turn, is based on the classic argument in the original Black–Scholes paper;
- The price of the underlying asset (typically a stock) follows a geometric Brownian motion, that is

$$\frac{dS}{S} = \mu\, dt + \sigma\, dW$$

where W is a stochastic variable.

Derivation of the Black–Scholes PDE

- The price of an Option $U(S,t)$ is a function of S and t, a simple application of Ito's lemma gives us

$$dU(S,t) = \left(\mu S \frac{\partial U(S,t)}{\partial S} + \frac{\partial U(S,t)}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U(S,t)}{\partial S^2}\right) dt$$
$$+ \sigma S \frac{\partial U(S,t)}{\partial S}\, dW$$

- Now consider a certain portfolio, called the delta-hedge portfolio, consisting of being short one option and long $\frac{\partial U(S,t)}{\partial S}$ shares at time t. The value of these holdings is

$$\Pi = -U(S,t) + \frac{\partial U(S,t)}{\partial S} S$$

- Over the time period $[t, t + dt]$, the total profit or loss from changes in the values of the holdings is

$$d\Pi = -dU(S,t) + \frac{\partial U(S,t)}{\partial S} dS$$

Derivation of the Black–Scholes PDE

- Using the previous equations we can write

$$
\begin{aligned}
d\Pi = & -\left(\mu S \frac{\partial U(S,t)}{\partial S} + \frac{\partial U(S,t)}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U(S,t)}{\partial S^2}\right) dt \\
& - \sigma S \frac{\partial U(S,t)}{\partial S} dW + \frac{\partial U(S,t)}{\partial S} (\mu\, S\; dt + \sigma\, S\, dW) \\
= & -\left(\frac{\partial U(S,t)}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U(S,t)}{\partial S^2}\right) dt
\end{aligned}
$$

  Notice that the ΔW term has vanished;
- Thus uncertainty has been eliminated and the portfolio is effectively riskless.
- The rate of return on this portfolio must be equal to the rate of return on any other riskless instrument; otherwise, there would be opportunities for arbitrage.

Derivation of the Black–Scholes PDE

- Now assuming the risk-free rate of return is r we must have over the time period $[t, t + \Delta t]$

$$r\Pi\,\Delta t = \Delta\Pi$$

- If we now equate our two formulas for $\Delta\Pi$ we obtain:

$$\left(-\frac{\partial U(S,t)}{\partial t} - \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U(S,t)}{\partial S^2}\right)\Delta t = r\left(-U(S,t) + S\frac{\partial U(S,t)}{\partial S}\right)\Delta t$$

Simplifying, we arrive at the celebrated Black–Scholes partial differential equation:

$$\boxed{\frac{\partial U(S,t)}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 U(S,t)}{\partial S^2} + rS\frac{\partial U(S,t)}{\partial S} - rV = 0}$$

Derivation of the Black–Scholes PDE

- Remember the general form of a linear second order PDE:

$$Au_{xx} + 2Bu_{xy} + Cu_{yy} + Du_x + Eu_y + F = 0$$

- If we assume $x = t$ and $y = S$ we easily find that:

$$B = 0, \quad A = 0 \Rightarrow B^2 - AC = 0$$

the B and S equation is trivially a parabolic PDE

Table of Contents

Basic Ideas

- In numerical analysis, **finite-difference methods** are discretization methods used for solving **differential equations** by approximating them with difference equations in which finite differences approximate the derivatives;
- In general, finite difference methods are used to price options by approximating the (continuous-time) differential equation that describes how an option price evolves over time by a set of (discrete-time) difference equations obtained by approximation of the partial derivative

$$\frac{\partial C}{\partial S} \Rightarrow \frac{\Delta C}{\Delta S} \tag{3}$$

- The discrete difference equations may then be solved iteratively to calculate a price for the option;

Basic Ideas

- Let's start from a very simple case;
- The problem is described by a simple *ODE (Ordinary Differential Equation)* of 2nd degree:

$$\frac{d^2u(t)}{dt^2} + \omega^2 u(t) = 0,\ u(0) = I,\ u'(0) = 0,\ t \in (0, T] \tag{4}$$

- We want to discover the solution using a numerical approach.
- The equation in of second degree so we need two initial condition, the first one could be the value of u the beginning $(t = 0)$, the second usually describe a condition on the first derivative. Intuitively, we might correctly expect that two conditions are sufficient, considering the fact that you could integrate this equation twice and this will deliver two constant of integration.

Basic Ideas: The Finite Difference Scheme

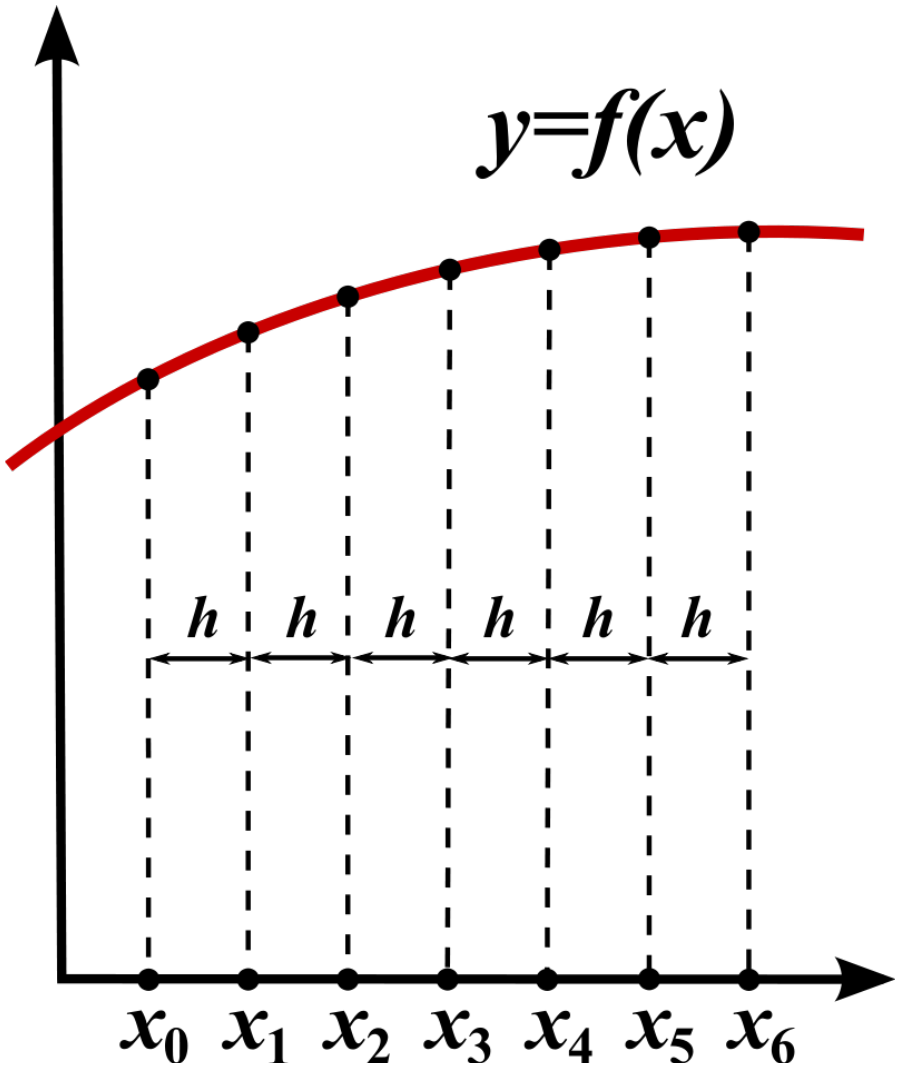
To formulate a finite difference method for the model problem we follow the following steps

- Discretizing the domain
- Fulfilling the equation at discrete time points
- Replacing derivatives by finite differences
- Dealing with Initial and Boundary Conditions
- Formulating a recursive algorithm

Basic Ideas: Discretizing the Domain

- The domain is discretized by introducing a uniformly partitioned time mesh.
- The points in the mesh are $t_n = n\Delta t$, $n = 0, 1, \ldots, N_t$, where $\Delta t = T/N_t$ is the constant length of the time steps.
- We introduce a mesh function u_n for $n = 0, 1, ..., N_t$, which approximates the exact solution at the mesh points. (Note that $n = 0$ is the known initial condition, so u_n is identical to the mathematical u at this point.)
- The mesh function un will be computed from algebraic equations derived from the differential equation problem.

Basic Ideas: Discretizing the Domain



Basic Ideas: Fulfilling the Equation at Discrete Time Points

- The ODE (Ordinary Differential Equation) is to be satisfied at each mesh point where the solution must be found:

$$u''(t_n) + \omega^2 u(t_n) = 0, \quad n = 1, ..., N_t \tag{5}$$

Basic Ideas: Replacing Derivatives by Finite Differences

- The derivative $u''(t_n)$ is to be replaced by a finite difference approximation.
- A common second-order accurate approximation to the second-order derivative is

$$f''(x) \sim \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

- Using this we obtain

$$\frac{u_{n+1} - 2u_n + u_{n-1}}{\Delta t^2} = -\omega^2 u_n \tag{6}$$

Basic Ideas: Initial Conditions

- The *initial condition* for the first derivative is simply $u' = 0$, we also need to replace the derivative in the initial condition by a finite difference. Here we choose another centered difference approximation

$$f'(x) \sim \frac{f(x+h) - f(x-h)}{2h} \Rightarrow \frac{u[1] - u[-1]}{2\Delta t} = 0$$

Basic Ideas: Recursive Algorithm

- To formulate the computational algorithm, we assume that we have already computed u_{n-1} and u_n, such that u_{n+1} is the unknown value to be solved for;
- The ordinary differential equation becomes

$$\frac{u[n+1] - 2u[n] + u[n-1]}{(\Delta t)^2} = -\omega^2 \, u[n]$$

or

$$u[n+1] = 2\,u[n] - u[n-1] - (\Delta t)^2 \,\omega^2 \,u[n]$$

Basic Ideas: Recursive Algorithm

$$\boxed{u[n+1] = 2\,u[n] - u[n-1] - (\Delta t)^2\,\omega^2\,u[n]}$$

We observe that this equation cannot be used for $n = 0$ since we have an undefined value of $u[-1]$ but the condition on the first derivative can help us because $u' = 0$ implies $u[1] = u[-1]$ so we have for $n = 0$

$$u[1] = u[0] - \frac{1}{2}\,(\Delta t)^2\,\omega^2\,u[0]$$

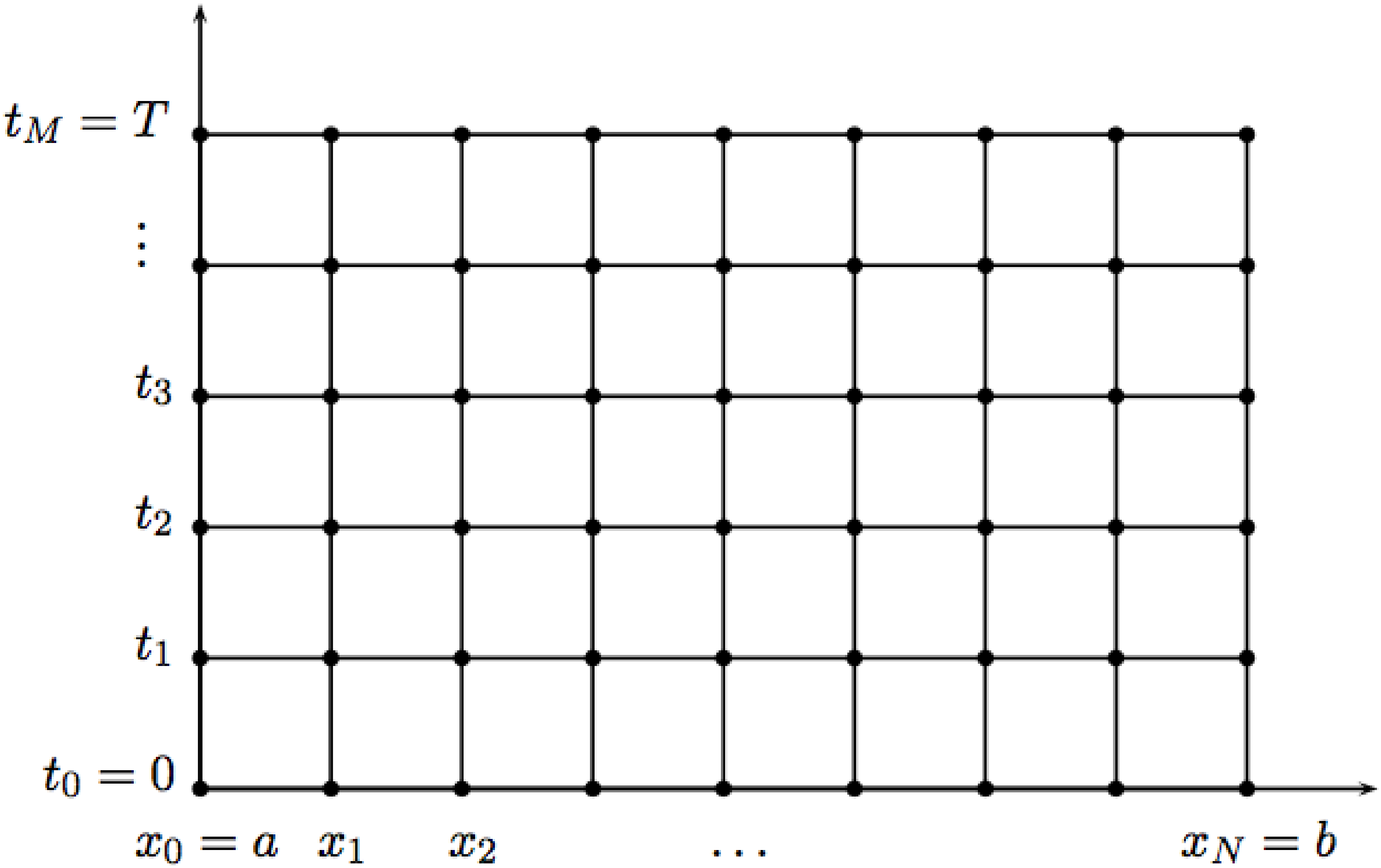
We have transformed our problem in an iterative equation (a **finite difference equation** that starting from value at steps n and $n-1$ gives us the value of our function to the next time step

Let's code ...

Table of Contents

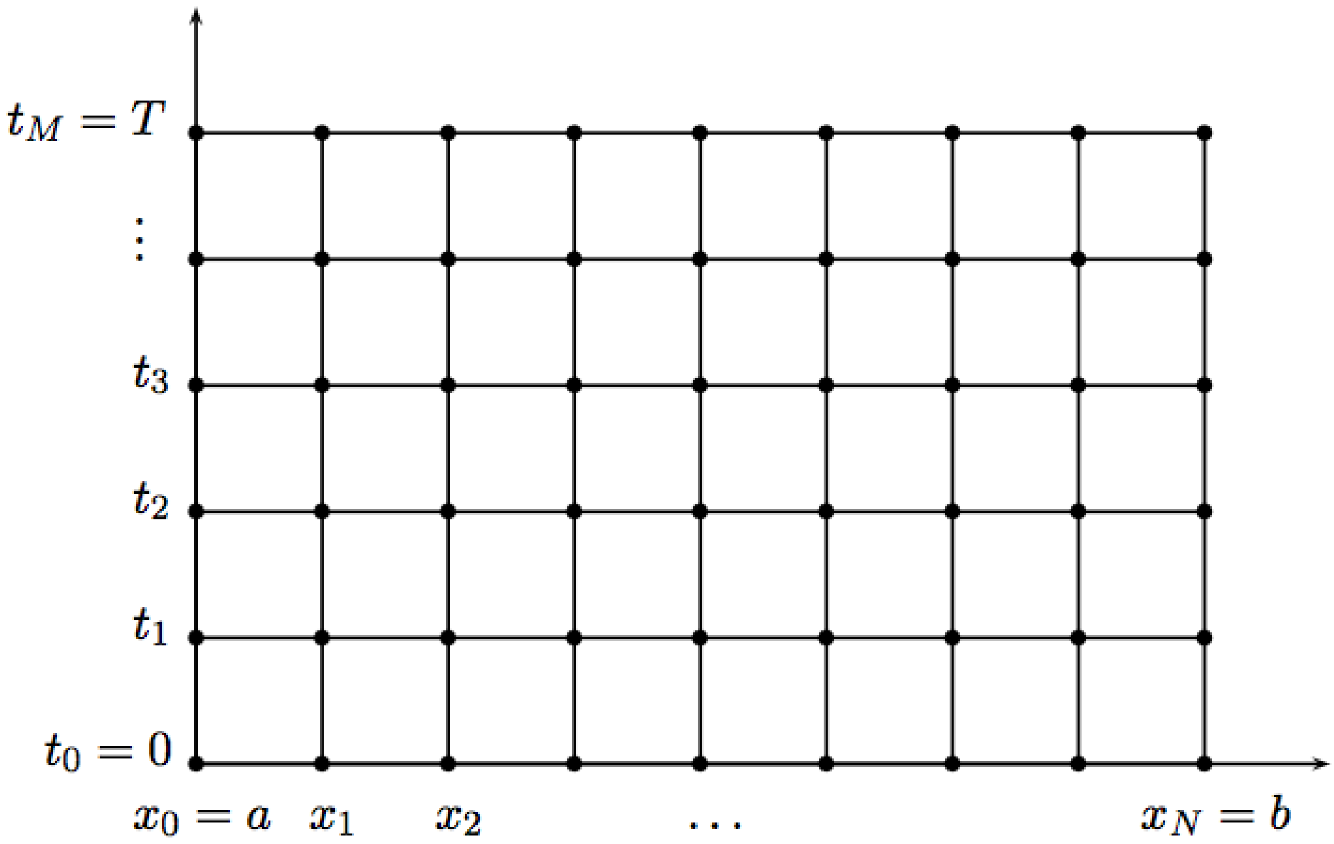
Building a Uniform Grid

The stock-time (or S-t) plane is discretized into a grid of nodes.



Building a Uniform Grid

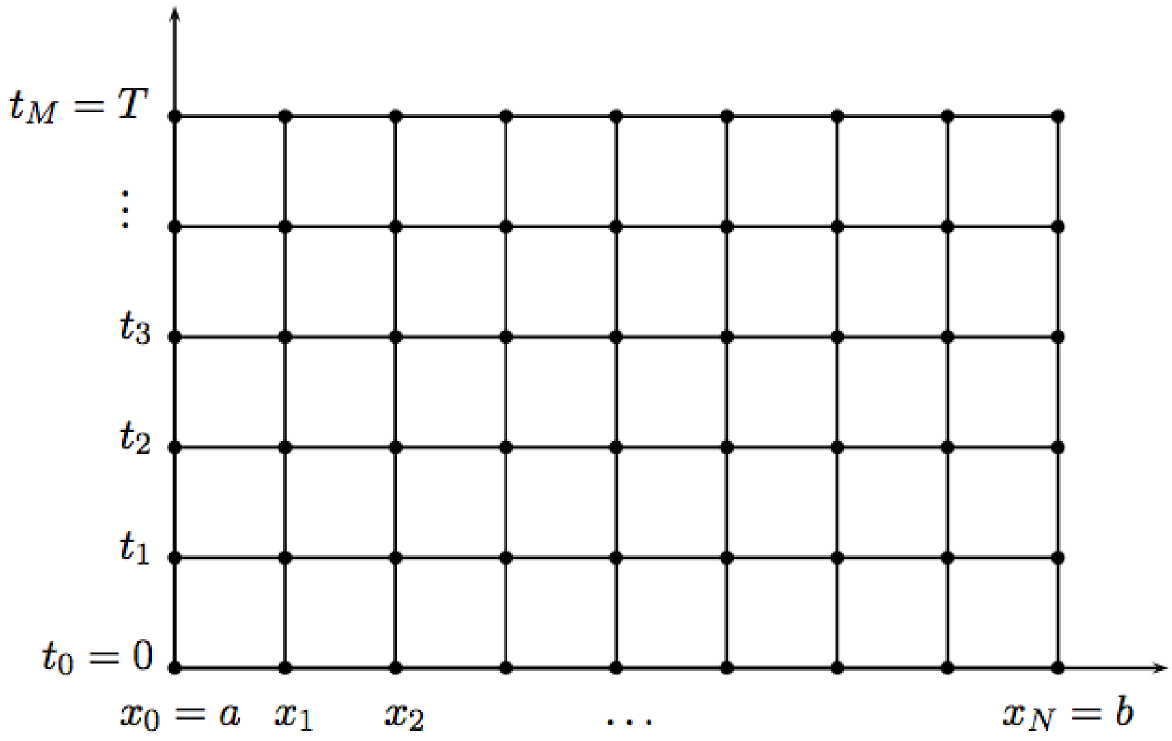
The examined time points are $0, \Delta t, 2\Delta t, \ldots, K\Delta t$, and the examined stock prices are $0, \Delta S, 2\Delta S, \ldots, I\Delta S$. $0 \leq i \leq I$ and $0 \leq k \leq K$



Building a Uniform Grid

The numerical solution of f means to find the values for all $U_{i,k}$, where i and k are the indexes for the Stock price and the Time, respectively:

$$S = i\Delta S, \qquad t = T - k\Delta T$$



Discrete Approximation for Derivative

- **Forward Approximation (1st order)**

$$f'(x) = \frac{f(x+h) - f(x)}{h} + O(h)$$

- **Backward Approximation (1st order)**

$$f'(x) = \frac{f(x) - f(x-h)}{h} + O(h)$$

- **Central Approximation (1st order)**

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} + O(h^2)$$

Discrete Approximation for Derivative

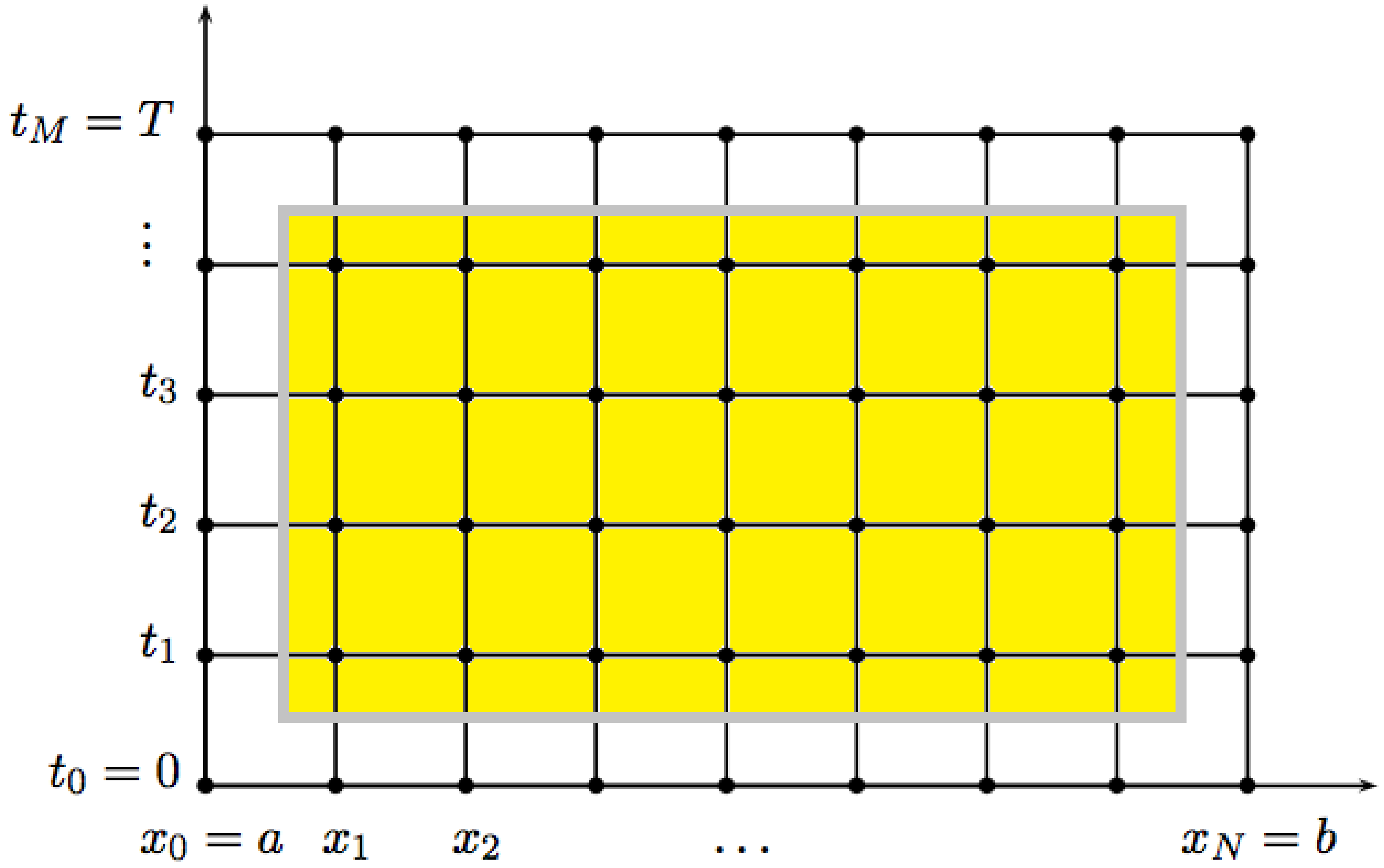
- From Taylor's series we can obtain a valid approximation for the second derivative up to third order terms

$$
\begin{aligned}
f(x+h) &= f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 + O(x^3) \\
f(x-h) &= f(x) - f'(x)h + \frac{1}{2}f''(x)h^2 + O(x^3)
\end{aligned}
\tag{7}
$$

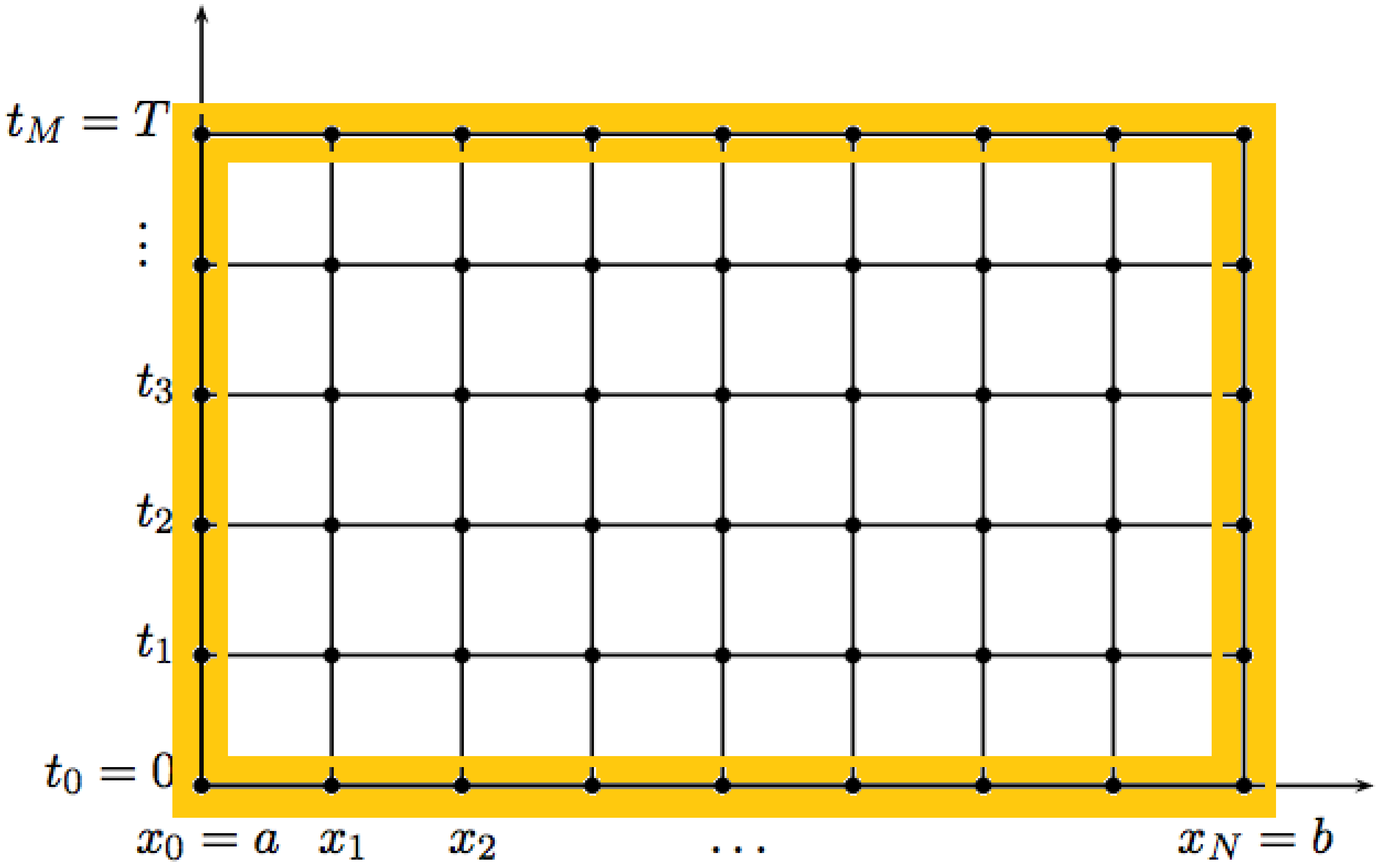
summing up we obtain

$$
f''(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} + O(x^3) \tag{8}
$$

Final and Boundary Conditions



Final and Boundary Conditions



Final Conditions and Payoff

- We know that at expiry the option value is just the payoff function.
- This means that we don't have to solve anything for time T.
- At expiry we have

$$U(S,t) = \text{Payoff}(S) \Rightarrow U_{i,0} = \text{Payoff}(i\Delta S)$$

- The right-hand side is a known function. For example, if we are pricing a call option we put

$$U_{i,0} = \max(i\Delta S - K, 0)$$

- This final condition will get our finite-difference scheme started. It will be just like working down the tree in the binomial model.

Boundary Condition

- We must specify the solution value at the extremes of the region.
- In our specific pricing problem this means that we have to prescribe the option value at $S = 0$ and $S = I\Delta S$.
- What we specify will depend on the type of option we are solving. Let's see some examples...

Boundary Condition: Call Option

- At $S = 0$ we have always

$$U_{0,k} = 0 \tag{9}$$

- For large value of S the call value approximate $S - Ke^{-r(T-t)}$ so we can set

$$U_{I,k} = I\Delta S - Ke^{-rk\Delta t} \tag{10}$$

Boundary Condition: Put Option

- For a put option we have the condition at $S = 0$ that $P = Ke^{-r(T-t)}$, this becomes

$$U_{0,k} = Ke^{-rk\Delta t} \tag{11}$$

- The put option become worthless for large S so

$$U_{I,k} = 0 \tag{12}$$

Boundary Condition: Linear Payoff

- When the option has a payoff that is most linear in the underlying for large value of S then you can use the upper boundary condition

$$\frac{\partial^2 U(S,t)}{\partial S^2} \to 0 \quad \text{as} \quad S \to \infty$$

- The finite-difference representation is

$$U_{I,k} - 2U_{I-1,k} + U_{I-2,k} = 0 \Rightarrow U_{I,k} = 2U_{I-1,k} - U_{I-2,k} \tag{13}$$

Table of Contents

Solving Black and Scholes PDE

Our goal is to solve the well known Black andScholes PDE

$$\frac{\partial}{\partial t}U(S,t)+\frac{1}{2}\sigma^2 S^2\cdot\frac{\partial^2}{\partial S^2}U(S,t)+rS\cdot\frac{\partial}{\partial S}U(S,t)-r\cdot U(S,t)=0$$

or, more generally:

$$\frac{\partial}{\partial t}U(S,t)+a(S)\cdot\frac{\partial^2}{\partial S^2}U(S,t)+b(S)\cdot\frac{\partial}{\partial S}U(S,t)+c(S)\cdot U(S,t)=0$$

Solving Black and Scholes PDE

$$\frac{\partial}{\partial t}U(S,t) \longrightarrow \frac{U_{i,k} - U_{i,k+1}}{\Delta t}$$
$$\frac{\partial}{\partial S}U(S,t) \longrightarrow \frac{U_{i+1,k} - U_{i-1,k}}{2\Delta S}$$
$$\frac{\partial^2}{\partial S^2}U(S,t) \longrightarrow \frac{U_{i+1,k} - 2U_{i,k} + U_{i-1,k}}{\Delta S^2}$$

Note that for the time derivative we choose the **backward difference** at time n, while for the space derivative we choose the central difference:

Solving Black and Scholes PDE

$$\frac{U_{i,k} - U_{i,k+1}}{\Delta t} + a\frac{U_{i+1,k} - 2U_{i,k} + U_{i-1,k}}{\Delta S^2} + b\frac{U_{i+1,k} - U_{i-1,k}}{2\Delta S} + cU_{i,k} = 0$$

or

$$\frac{U_{i,k+1} - U_{i,k}}{\Delta t} = a\frac{U_{i+1,k} - 2U_{i,k} + U_{i-1,k}}{\Delta S^2} + b\frac{U_{i+1,k} - U_{i-1,k}}{2\Delta S} + cU_{i,k}$$

Solving Black and Scholes PDE

$$
\begin{aligned}
U_{i,k+1} &= a\Delta t \frac{U_{i+1,k} - 2U_{i,k} + U_{i-1,k}}{\Delta S^2} + b\Delta t \frac{U_{i+1,k} - U_{i-1,k}}{2\Delta S} + \Delta t c U_{i,k} + U_{i,k} \\
&= \frac{a\Delta t}{\Delta S^2} U_{i+1,k} + \frac{b\Delta t}{2\Delta S} U_{i+1,k} - 2\frac{a\Delta t}{\Delta S^2} U_{i,k} + \Delta t c U_{i,k} + U_{i,k} \\
&+ \frac{a\Delta t}{\Delta S^2} U_{i-1,k} - \frac{b\Delta t}{2\Delta S} U_{i-1,k}
\end{aligned}
$$

$$
\begin{aligned}
U_{i,k+1} = U_{i+1,k} &\left[\frac{a\Delta t}{\Delta S^2} + \frac{b\Delta t}{2\Delta S}\right] + U_{i,k}\left[1 + \Delta t c - 2\frac{a\Delta t}{\Delta S^2}\right] \\
&+ U_{i-1,k}\left[\frac{a\Delta t}{\Delta S^2} - \frac{b\Delta t}{2\Delta S}\right]
\end{aligned}
$$

$$\boxed{U_{i,k+1} = A\,U_{i+1,k} + B\,U_{i,k} + C\,U_{i-1,k}}$$

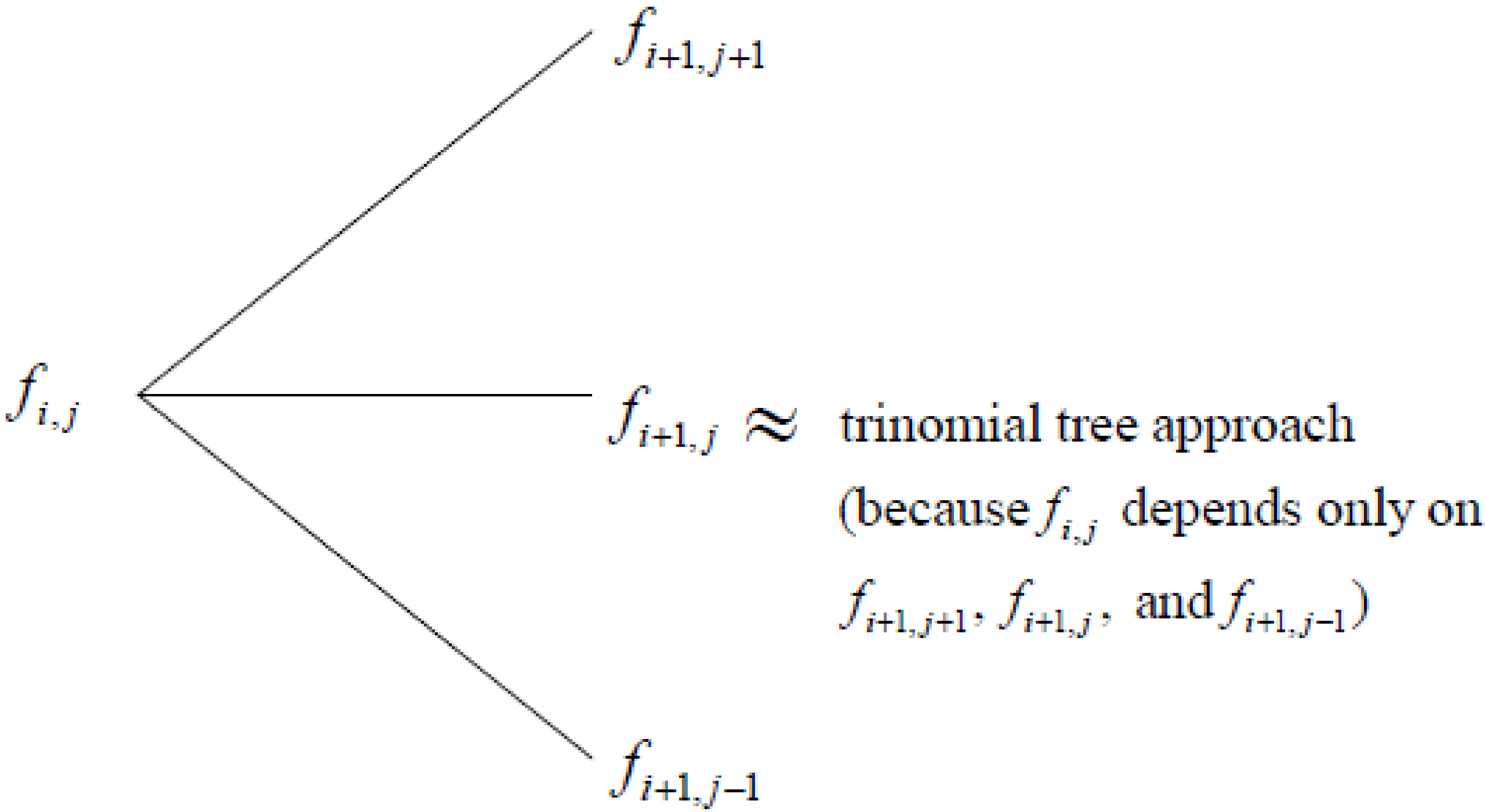
with

$$\begin{aligned}
A &= \frac{\Delta t}{\Delta S^2} a + \frac{\Delta t}{2\Delta S} b = \nu_1 a + \frac{1}{2}\nu_2 b \\
B &= 1 + \Delta t\, c - \frac{2\Delta t}{\Delta S^2} a = 1 - 2\nu_1 a + \Delta t\, c \\
C &= \frac{\Delta t}{\Delta S^2} a - \frac{\Delta t}{2\Delta S} b = \nu_1 a - \frac{1}{2}\nu_2 b
\end{aligned}$$

Let's code ...

Equivalence with Trinomial Trees

$$\boxed{U_{i,k+1} = A\,U_{i+1,k} + B\,U_{i,k} + C\,U_{i-1,k}}$$



Equivalence with Trinomial Trees

Let's start from a slightly different version of the discrete equation

$$\frac{U_{i,k+1} - U_{i,k}}{\Delta t} = a\frac{U_{i+1,k} - 2U_{i,k} + U_{i-1,k}}{\Delta S^2} + b\frac{U_{i+1,k} - U_{i-1,k}}{2\Delta S} + cU_{i,k+1}$$

$$\begin{aligned} U_{i,k+1}\left(1 - c\Delta t\right) = U_{i+1,k} &\left[\frac{a\Delta t}{\Delta S^2} + \frac{b\Delta t}{2\Delta S}\right] \\ + U_{i,k} &\left[1 - 2\frac{a\Delta t}{\Delta S^2}\right] \\ + U_{i-1,k} &\left[\frac{a\Delta t}{\Delta S^2} - \frac{b\Delta t}{2\Delta S}\right] \end{aligned}$$

Equivalence with Trinomial Trees

$$U_{i,k+1} = AU_{i+1,k} + BU_{i,k} + CU_{i-1,k} \tag{14}$$

with

$$A = \frac{1}{1 - c\Delta t}\left[\frac{\Delta t}{\Delta S^2}a + \frac{\Delta t}{2\Delta S}b\right]$$
$$B = \frac{1}{1 - c\Delta t}\left[1 - \frac{2\Delta t}{\Delta S^2}a\right]$$
$$C = \frac{1}{1 - c\Delta t}\left[\frac{\Delta t}{\Delta S^2}a - \frac{\Delta t}{2\Delta S}b\right]$$

Equivalence with Trinomial Trees

Substitution for a, b, c values give us (remember that due to the discretization process we have also $S = i\Delta S$ and $\Delta t = k$)

$$A = \frac{1}{1+r\Delta t}\left[\frac{\Delta t}{2\Delta S^2}\sigma^2 S^2 + \frac{\Delta t}{2\Delta S} rS\right] = \frac{1}{1+r\Delta t}\left[\frac{1}{2}\sigma^2 i^2 \Delta t + \frac{1}{2} ri\Delta t\right] = \frac{1}{1+r\Delta t} A^\star$$

$$B = \frac{1}{1+r\Delta t}\left[1 - \frac{\Delta t}{\Delta S^2}\sigma^2 S^2\right] = \frac{1}{1+r\Delta t}\left[1 - \sigma^2 i^2 \Delta t\right] = \frac{1}{1+r\Delta t} B^\star$$

$$C = \frac{1}{1+r\Delta t}\left[\frac{\Delta t}{2\Delta S^2}\sigma^2 S^2 - \frac{\Delta t}{2\Delta S} rS\right] = \frac{1}{1+r\Delta t}\left[\frac{1}{2}\sigma^2 i^2 \Delta t - \frac{1}{2} ri\Delta t\right] = \frac{1}{1+r\Delta t} C^\star$$

Now the relation with a trinomial tree is completely evident. Note that

$$A^\star + B^\star + C^\star = 1$$

Equivalence with Trinomial Trees

We can interpret terms as follows

- $\frac{1}{2}\sigma^2 i^2 \Delta t + \frac{1}{2} r i \Delta t$: probability of stock price increasing from $i\Delta S$ to $(i+1)\Delta S$ in time Δt;
- $1 - \sigma^2 i^2 \Delta t$: probability of stock price remaining unchanged at $i\Delta S$ in time Δt;
- $\frac{1}{2}\sigma^2 i^2 \Delta t - \frac{1}{2} r i \Delta t$: probability of stock price decreasing from $i\Delta S$ to $(i+1)\Delta S$ in time Δt;

The value at time $k+1$ is simply the discounted value of the expectation a time k.

$$\boxed{U_{i,k+1} = \frac{1}{1 + r\Delta t}\left(A^\star U_{i+1,k} + B^\star U_{i,k} + C^\star U_{i-1,k}\right)}$$

Convergence of the Explicit Method

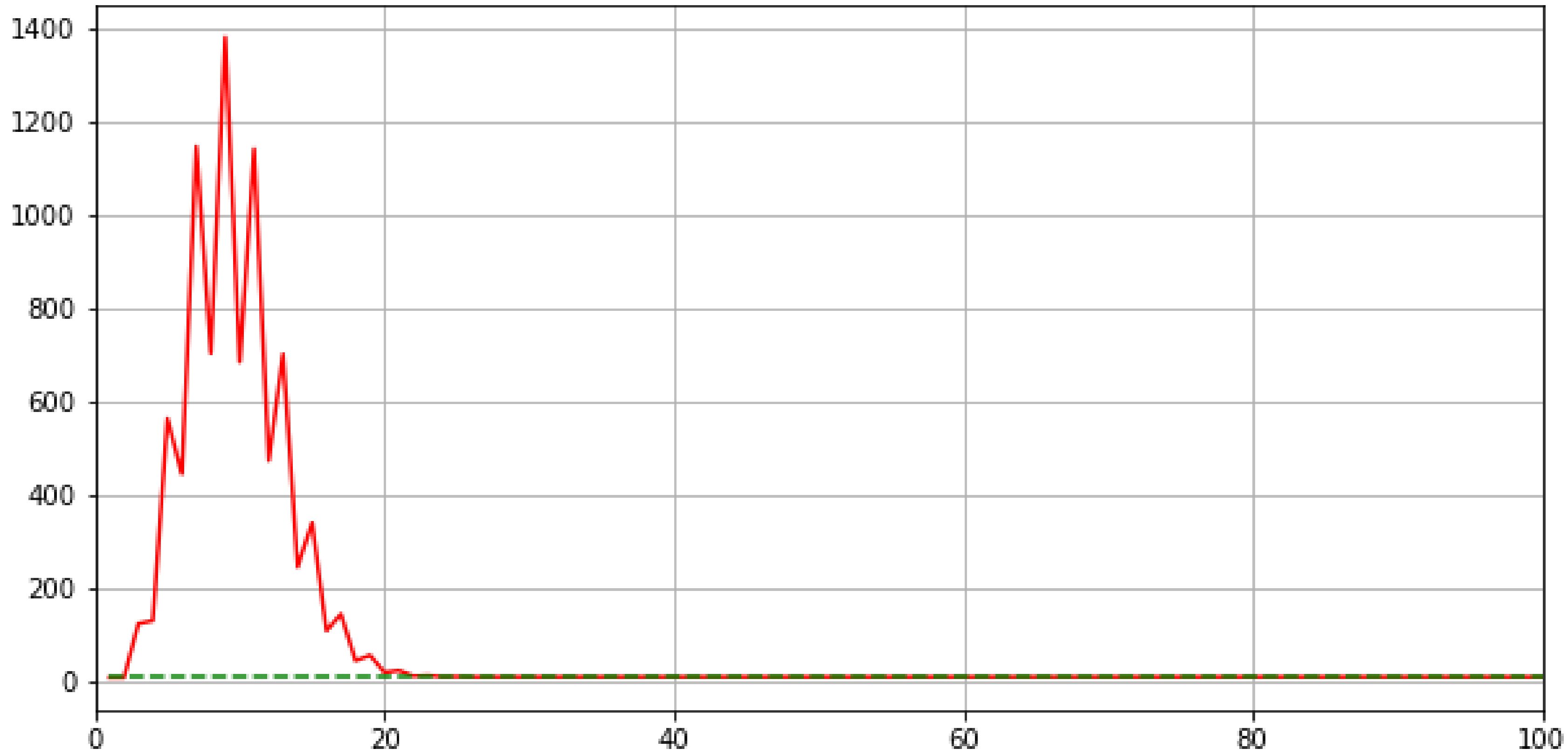
- Although the explicit method is simple to implement it does not always converge.
- Convergence of the method depends on the size of the time step, the size of the asset step and the size of coefficients a, b and c.
- Tipically there is a severe limitation on the size of the time step

$$\Delta t \leq \frac{\Delta S^2}{2a}$$

- if we want to improve accuracy by halving the asset step, for example, we must reduce the timestep by a factor of four. The computation time then goes up by a factor of eight.

Convergence of the Explicit Method

If the time step constraint is not satisfied, if it is too large, then the instability is obvious from the results. It is unlikely that you will get a false but believable result if you use the explicit method.



Convergence of the Explicit Method

This behaviour can be understood very well by the analogy with the trinomial tree. Infact since all terms can be interpreted as probability, we must impose

$$1 - \sigma^2 i^2 \Delta t \geq 0 \Rightarrow \sigma^2 i^2 \Delta t \leq 1$$

from this taking into account that $S = i\Delta S$,

$$\Delta t \leq \frac{1}{\sigma^2 S^2} \Delta S^2$$

And from the third probability

$$\frac{1}{2}\sigma^2 i^2 \Delta t - \frac{1}{2} r i \Delta t \geq 0$$

we can easily find a constraint on the step size of S

$$\Delta S \leq \frac{\sigma^2 S}{r}$$

Table of Contents

Implicit Finite-Difference Method

- The **implicit method** is superficially just like the explicit scheme but now the delta and gamma (first and second derivative with respect to S) are calculated at time step $k+1$ instead of k

$$\frac{U_{i,k}-U_{i,k+1}}{\Delta t}+a_i\left(\frac{U_{i+1,k+1}-2U_{i,k+1}+U_{i-1,k+1}}{\Delta S^2}\right)+$$
$$b_i\left(\frac{U_{i+1,k+1}-U_{i-1,k+1}}{2\Delta S}\right)+c_iU_{i,k+1}=0$$

$$\boxed{A_iU_{i+1,k+1}+B_iU_{i,k+1}+C_iU_{i-1,k+1}=U_{i,k}}$$

Implicit Finite-Difference Method

$$\boxed{A_i U_{i+1,k+1} + B_i U_{i,k+1} + C_i U_{i-1,k+1} = U_{i,k}}$$

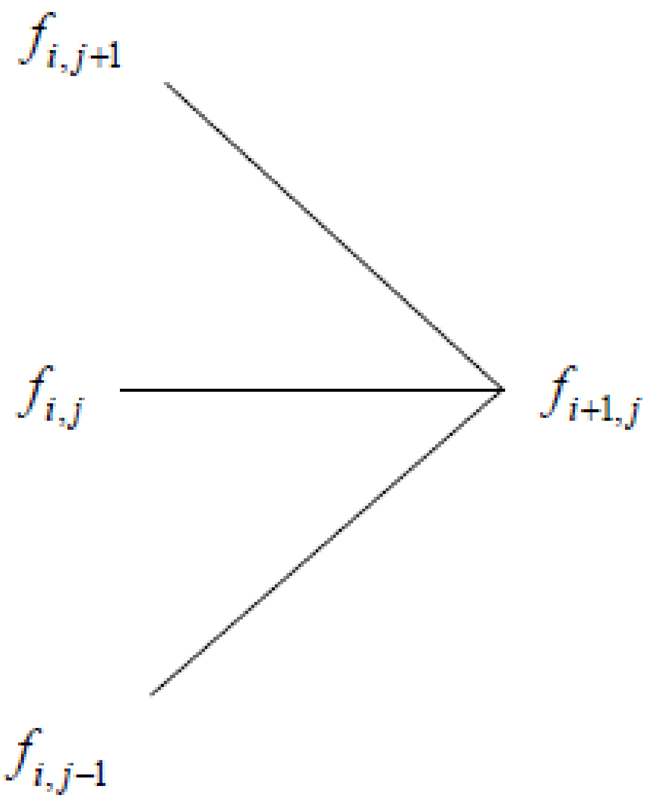
$$A_i = -\nu_1 a_i + \frac{1}{2}\nu_2 b_i$$
$$B_i = 1 + 2\nu_1 a_i - \Delta t\, c_i$$
$$C_i = -\nu_1 a_i - \frac{1}{2}\nu_2 b_i$$

where

$$\nu_1 = \frac{\Delta t}{\Delta S^2}, \qquad \nu_2 = \frac{\Delta t}{\Delta S}$$

Implicit Finite-Difference Method

$$\boxed{A_i U_{i+1,k+1} + B_i U_{i,k+1} + C_i U_{i-1,k+1} = U_{i,k}}$$



$f_{i+1,j} \approx$ multinomial tree approach (because $f_{i,j}$ depends on the values of $f_{i+1,m}, \cdots, f_{i+1,0}$)

Implicit Finite-Difference Method

- These scheme is very different from the explicit scheme, first of all it's possible to demonstrate that the method doesn't suffer anymore from the restriction on the time step.
- The asset step can be small and the time step large without the method running into stability problem.
- The second difference concern the solution procedure, infact the solution of the difference equation is no longer so straightforward.
- To get $U_{i,k+1}$ from $U_{i,k}$ we have to solve a set of linear equations, each $U_{i,k+1}$ is directly linked to its two neighbours and thus indirectly linked to every option value at the same time step.

Implicit Finite-Difference Method

$$
\begin{bmatrix}
A_1 & 1+B_1 & C_1 & 0 & \cdots & 0 \\
0 & A_2 & 1+B_2 & C_2 & \cdots & 0 \\
0 & 0 & A_3 & 1+B_3 & \cdots & 0 \\
\vdots & & \ddots & \ddots & \ddots & \vdots \\
0 & \cdots & & A_{I-1} & 1+B_{I-1} & C_{I-1}
\end{bmatrix}
\cdot
\begin{bmatrix}
U_{0,k+1} \\ U_{1,k+1} \\ U_{2,k+1} \\ \vdots \\ U_{I-1,k+1} \\ U_{I,k+1}
\end{bmatrix}
$$

$$
= \begin{bmatrix}
U_{0,k} \\ U_{1,k} \\ U_{2,k} \\ \vdots \\ U_{I-1,k} \\ U_{I,k}
\end{bmatrix}
$$

This matrix has $I-1$ rows and $I+1$ columns so this is a representation of $I-1$ equations in $I+1$ unknowns. The two equations that we are missing come from the boundary conditions.

Implicit Finite-Difference Method

$$
\begin{bmatrix}
1+B_1 & C_1 & 0 & \cdots \\
A_2 & 1+B_2 & C_2 & \cdots \\
0 & A_3 & 1+B_3 & \cdots \\
\ddots & \ddots & \ddots & \vdots \\
\cdots & \cdots & A_{I-1} & 1+B_{I-1}
\end{bmatrix}
\cdot
\begin{bmatrix}
U_{1,k+1} \\
U_{2,k+1} \\
\vdots \\
U_{I-1,k+1}
\end{bmatrix}
$$

$$
+\begin{bmatrix}
A_1 U_{0,k+1} - U_{0,k} \\
0 \\
\vdots \\
C_I U_{I,k+1} - U_{I,k}
\end{bmatrix}
=
\begin{bmatrix}
U_{1,k} \\
U_{2,k} \\
\vdots \\
U_{I-1,k}
\end{bmatrix}
$$

Implicit Finite-Difference Method

$$\begin{bmatrix} 1+B_1 & C_1 & 0 & \cdots \\ A_2 & 1+B_2 & C_2 & \cdots \\ 0 & A_3 & 1+B_3 & \cdots \\ \ddots & \ddots & \ddots & \vdots \\ \cdots & \cdots & A_{I-1} & 1+B_{I-1} \end{bmatrix} \cdot \begin{bmatrix} U_{1,k+1} \\ U_{2,k+1} \\ \vdots \\ U_{I-1,k+1} \end{bmatrix}$$

$$= \begin{bmatrix} U_{1,k} \\ U_{2,k} \\ \vdots \\ U_{I-1,k} \end{bmatrix} - \begin{bmatrix} A_1 U_{0,k+1} - U_{0,k} \\ 0 \\ \vdots \\ C_I U_{I,k+1} - U_{I,k} \end{bmatrix} = \begin{bmatrix} q_{1,k} \\ q_{2,k} \\ \vdots \\ q_{I-1,k} \end{bmatrix}$$

Implicit Finite-Difference Method

$$\boxed{\mathbf{M_{imp}U = q}}$$

- This matrix equation holds whichever of the boundary conditions we have.
- Solution methods are various but two of the most used are **LU Decomposition** and **Successive Over Relaxation**.
- The LU decomposition is an example of a 'direct method'. This means that it finds the exact solution of the equations in one go through matrix operations.

Implicit Finite-Difference Method

$$\boxed{\mathbf{M_{imp} U = q}}$$

- An alternative strategy is to employ an **iterative method**.
- Iterative methods differ from direct methods in that one starts with a guess for the solution and successively improves it until it converges near enough to the exact solution.
- In a direct method one obtains the solution without any interaction.

An advantage of iterative methods over direct methods is that they generalise in straightforward ways to American option problems whereas direct methods do not

Table of Contents

Successive Over-Relaxation Method

Suppose that the **square** matrix $\mathbf{M}$ in the matrix equation

$$\mathbf{M}\mathbf{v} = \mathbf{q}$$

has entries M_{ij} then the system of equations can be written as

$$\begin{aligned}
&M_{11}v_1 + M_{12}v_2 + \cdots + M_{1N}v_N = q_1 \\
&M_{21}v_1 + M_{22}v_2 + \cdots + M_{2N}v_N = q_2 \\
&\ldots\ldots \\
&M_{N1}v_1 + M_{N2}v_2 + \cdots + M_{NN}v_N = q_N
\end{aligned}$$

N is the number of equations.

Successive Over-Relaxation Method

$$
\begin{aligned}
M_{11}v_1 &= q_1 - (M_{12}v_2 + \cdots + M_{1N}v_N) \\
M_{22}v_2 &= q_2 - (M_{21}v_1 + \cdots + M_{2N}v_N) \\
&\ldots\ldots \\
M_{NN}v_N &= q_N - (M_{N1}v_1 + \ldots)
\end{aligned}
$$

The system is easily solved *iteratively*

$$
\begin{aligned}
v_1^{n+1} &= \frac{1}{M_{11}}\left(q_1 - (M_{12}v_2^n + \cdots + M_{1N}v_N^n)\right) \\
v_2^{n+1} &= \frac{1}{M_{22}}\left(q_2 - (M_{21}v_1^n + \cdots + M_{2N}v_N^n)\right) \\
&\ldots\ldots \\
v_N^{n+1} &= \frac{1}{M_{NN}}\left(q_N - (M_{N1}v_1^n + \ldots)\right)
\end{aligned}
$$

Successive Over-Relaxation Method

- In this case, the superscript n denotes the level of the iteration, this iteration is started from some initial guess $\mathbf{v}^0$.
- In our case it is usual to start with the value of the option at the previous timestep as the initial guess for the next timestep.
- This iterative method is called the **Jacobi Method**

Successive Over-Relaxation Method

$$
\begin{aligned}
M_{11}v_1 &= q_1 - (\qquad\qquad\quad M_{12}v_2 + M_{13}v_3 + \dots + M_{1N}v_N) \\
M_{22}v_2 &= q_2 - (M_{21}v_1 \qquad\qquad\quad + M_{23}v_3 + \dots + M_{2N}v_N) \\
&\dots\dots \\
M_{NN}v_N &= q_N - (M_{N1}v_1 + M_{N2}v_2 + M_{N3}v_3 + \dots)
\end{aligned}
$$

Successive Over-Relaxation Method

$$
\begin{aligned}
M_{11}v_1 &= q_1 - (\qquad\qquad\quad \underbrace{M_{12}v_2 + M_{13}v_3 + \cdots + M_{1N}v_N)}\\
M_{22}v_2 &= q_2 - (M_{21}v_1 \qquad\qquad + M_{23}v_3 + \cdots + M_{2N}v_N)\\
&\cdots\cdots\\
M_{NN}v_N &= q_N - (M_{N1}v_1 + M_{N2}v_2 + M_{N3}v_3 + \ldots)
\end{aligned}
$$

$\mathbf{D} \cdot \mathbf{v}^{n+1}$

$\mathbf{L} \cdot \mathbf{v}^{n}$

$\mathbf{U} \cdot \mathbf{v}^{n}$

$$\boxed{\mathbf{D} \cdot \mathbf{v}^{n+1} = \mathbf{q} - \mathbf{U} \cdot \mathbf{v}^{n} - \mathbf{L} \cdot \mathbf{v}^{n}}$$

Successive Over-Relaxation Method

- The matrix $\mathbf{M}$ can be written as the sum of a diagonal matrix $\mathbf{D}$, an upper triangular matrix $\mathbf{U}$ with zeros on the diagonal and a lower triangular matrix $\mathbf{L}$ also with zeros on the diagonal:

$$\mathbf{M} = \mathbf{D} + \mathbf{U} + \mathbf{L}$$

Using this representation we can write the Jacobi method in a different but elegant way

$$\boxed{\mathbf{v}^{n+1} = \mathbf{D}^{-1}\left(\mathbf{q} - \mathbf{U}\cdot\mathbf{v}^{n} - \mathbf{L}\cdot\mathbf{v}^{n}\right)}$$

Successive Over-Relaxation Method

The idea of the **successive over-relaxation method** is to speed up convergence of the procedure using the right hand side of the previous equation as a correction factor of the value of $\mathbf{v}$ estimated at the previous step

$$\boxed{\mathbf{v}^{n+1} = \mathbf{v}^{n+1} + \omega \mathbf{D}^{-1} \left(\mathbf{q} - \mathbf{U} \cdot \mathbf{v}^{n} - \mathbf{L} \cdot \mathbf{v}^{n} \right)}$$

ω is the so called **over-relaxation** parameter and its value must lie between 1 and 2. Usually this parameter is estimated by experiment making some test for the optimal value.

Let's code ...

jupyter

Table of Contents

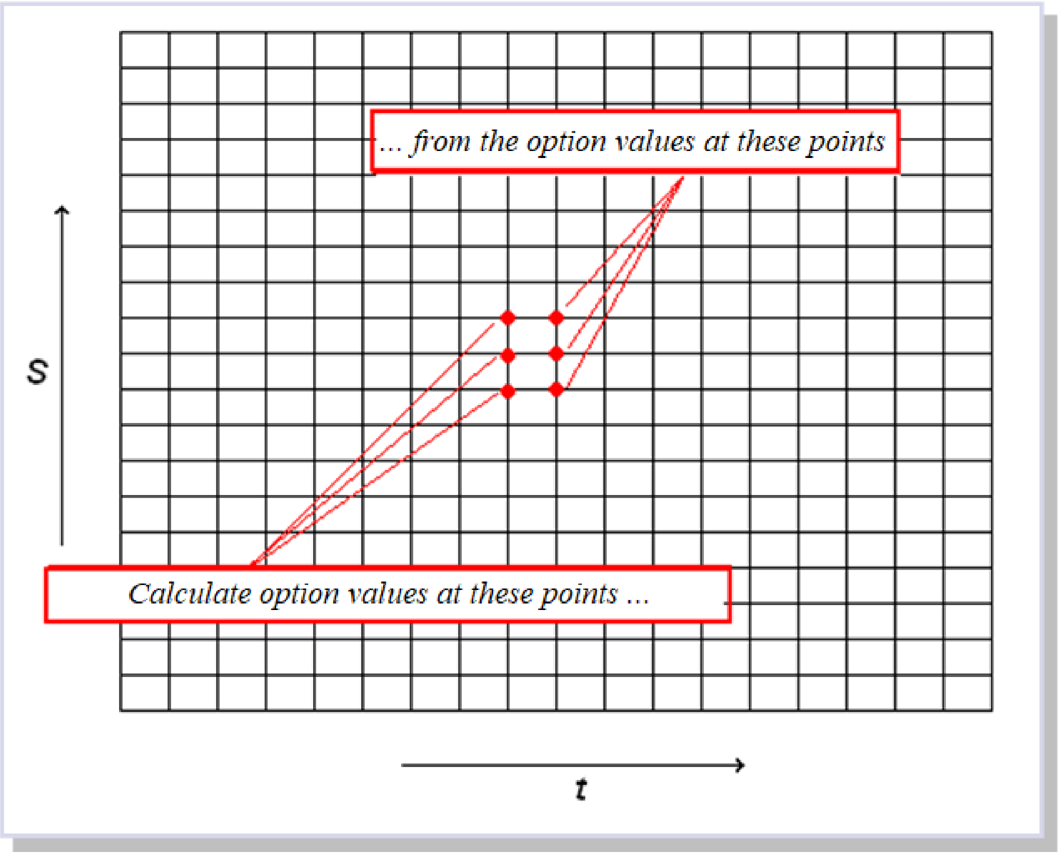
The Crank-Nicolson Method

- The Crank-Nicolson scheme is

$$
\begin{aligned}
&\frac{U_{i,k} - U_{i,k+1}}{\Delta t} + \frac{a_i}{2}\left(\frac{U_{i+1,k+1} - 2U_{i,k+1} + U_{i-1,k+1}}{\Delta S^2}\right) \\
&+ \frac{a_i}{2}\left(\frac{U_{i+1,k} - 2U_{i,k} + U_{i-1,k}}{\Delta S^2}\right) \\
&+ \frac{b_i}{2}\left(\frac{U_{i+1,k+1} - U_{i-1,k+1}}{2\Delta S}\right) + \frac{b_i}{2}\left(\frac{U_{i+1,k} - U_{i-1,k}}{2\Delta S}\right) \\
&+ \frac{1}{2}c_i U_{i,k} + \frac{1}{2}c_i U_{i,k+1} = 0
\end{aligned}
$$

The Crank-Nicolson Method

$$
\begin{aligned}
&-A_i U_{i-1,k+1} + (1 - B_i) U_{i,k+1} - C_i U_{i+1,k+1} = \\
&= A_i U_{i-1,k} + (1 + B_i) U_{i,k} + C_i U_{i+1,k}
\end{aligned}
$$



The Crank-Nicolson Method

- … see the notebook for more details