Numerical Methods in Option Pricing

Lesson VII: Numerical PDE resolution

Dr. Sebastian del Bano Rollin

Financial Computing and Analytics
Computer Science Department
University College London

4th March 2014

Outline I

Pricing Derivatives with PDEs

The PDEs

We know that to price financial derivatives we need to solve a PDE. This PDE describes the behaviour of the price of a derivative at time t and with spot at level S as a function $f(S,t)$.
This PDE is different according to the model for the dynamics of spot.
Some of the PDEs used are:

1. In the Black Scholes model the PDE is

BS PDE

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

2. In the Black Scholes model with a term structure of vol and rates the PDE is

TS BS PDE

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

Pricing Derivatives with PDEs

The PDEs

3. In a local volatility model the PDE is

LV PDE

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

for a function $\sigma(S,t)$ called the local volatility function that has to be inferred from the options vanilla market.

4. In a stochastic volatility model the function f depends on three variables: $f(S,\sigma,t)$ and the PDE is something like

SV PDE

$$\frac{\partial f}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 f}{\partial S^2} + \rho\sigma S b \frac{\partial^2 f}{\partial S \partial \sigma} + \frac{1}{2}b^2 \frac{\partial^2 f}{\partial \sigma^2} = r\left(f - S\frac{\partial f}{\partial S}\right)$$

Pricing Derivatives with PDEs
Boundary Conditions

In addition we have to add the boundary conditions to reflect the terms of the derivatives contract. For example:

1. For a call option struck at K expiring at time T with a Knock-Out barrier at B this is

$$\begin{aligned} f(S,T) &= (S-K)^+ && (1)\\ f(B,t) &= 0 \text{ for } 0 \le t \le T && (2) \end{aligned}$$

2. For a Double-No-Touch with barriers at B_L and B_U this is

$$\begin{aligned} f(S,T) &= 1 && (3)\\ f(B_L,t) &= f(B_U,t) = 0 \text{ for } 0 \le t \le T && (4) \end{aligned}$$

Pricing Derivatives with PDEs

Resolving financial PDEs

- It happens to be that most of these problems cannot be resolved exactly.
- For example, there is no closed form solution for the price of a DNT in a BS model with time varying vol and rate.
- We therefore require numerical methods to find approximate solutions.
  1. Numerical PDE methods.
  2. Monte Carlo methods.
  3. Sometimes we can find ad-hoc methods suited to a specific circumstance: e.g. numerical integrations, moment matching techniques,...

  The result linking PDE methods and MC methods is the Feynman-Kac Theorem that states that the solution to a certain PDE can be found by calculating an expectation.

Pricing Derivatives with PDEs
Rescaling

- All the PDEs we have described earlier have the term S^2 multiplying $\partial^2 f/\partial S^2 C$. This means that the PDE is not "space homogeneous".
- It would be preferable to make the PDE more spot homogeneous.
- To accomplish this we use the typical change of variables $s = \ln(S)$. This is "philosophically" the same as thinking of spot in percentage terms over a reference spot S_0.
- This converts the time dependent BS PDE into

TS BS PDE in log-spot

$$\frac{\partial f}{\partial t} + \frac{1}{2}\sigma(t)^2 \frac{\partial^2 f}{\partial s^2} = r(t) f - \left(r(t) - \frac{\sigma(t)^2}{2} \right) \frac{\partial f}{\partial s}$$

Pricing Derivatives with PDEs
Parabolic PDEs

- We are very lucky that the pricing PDEs are very similar to equations well known in mathematical physics.
- In the XIX century, Fourier showed how heat diffusion could be modelled by the equation

$$\frac{\partial u(x,t)}{\partial t} = a(x,t)\frac{\partial^2 u(x,t)}{\partial x^2}$$

  where $a(x,t)$ is a positive function related to the easiness for heat to diffuse in the given material.
- In higher dimensions $\partial^2 u(x,t)/\partial x^2$ is replaced by the Laplacian, which for $n=2$ is $\Delta u(x,y,t) = \partial^2 u(x,y,t)/\partial x^2 + \partial^2 u(x,y,t)/\partial y^2$.
- The heat equation is then $\partial u/\partial t = \Delta u$.
- Most of the properties of the heat equation are shared by equations called parabolic PDEs.

Parabolic PDEs

- On $\mathbb{R}^n$ a parabolic PDE is an equation of the form

$$\frac{\partial u}{\partial t} = \sum_{1\le i,j\le n} a_{ij}\frac{\partial^2 u}{\partial x_i \partial x_j} + \sum_{1\le i\le n} b_i \frac{\partial u}{\partial x_i} + cu + d$$

  where a_{ij}, b_i, c and d are functions of (x, t) and a_{ij} is a positive definite matrix.

- The condition that a_{ij} is positive definite is key to show that the PDE has solutions for certain boundary conditions. It is often called the ellipticity condition as if we remove the $\partial u/\partial t$ it is the condition for the resulting equation being *elliptic*.

Pricing Derivatives with PDEs

The Laplacian

- Using l'Hopital we see that the second derivative is the limit:

$$\frac{\partial u}{\partial x^2} = \lim_{\varepsilon \to 0} \frac{u(x-\varepsilon) - 2u(x) + u(x+\varepsilon)}{\varepsilon^2}.$$

- The formula above allows us to interpret $u''(x)$ as the "injustice" at x: $\partial^2 u / \partial x^2$ is twice the difference between the average value around us $\frac{1}{2}(u(x-\varepsilon) + u(x+\varepsilon))$ and the value at the point, $u(x)$ divided by ε^2.
  - If $u''(x) > 0$ then the function is on average larger around x.
  - If $u''(x) < 0$ then the function is on average smaller around x.
- The same is applicable in higher dimensions replacing $u''(x)$ by the Laplacian.

Pricing Derivatives with PDEs

Regularisation

- The heat equation modifies the function $u(x, t)$ as time passes by an amount proportional to the injustice (in the x axis):
  - If $\partial^2 u/\partial x^2 > 0$ then the function grows with time at x as $\partial u/\partial t = a\partial^2 u/\partial x^2 > 0$. This growth will correct the fact that $u(x)$ was on average lower than the value around x.
  - If $u''(x) < 0$ then the function decreases with time at x as $\partial u/\partial t = a\partial^2 u/\partial x^2 < 0$. This decrease will correct the fact that $u(x)$ was on average larger than the value around x.

  So it essentially averages out differences as time progresses.
- This effect is also described as regularising effect. As time passes the function u becomes more and more smooth.
- This is observed to be the case in heat diffusion and other diffusion processes.
- Note that in the reasoning above it is key that $a > 0$, in general this is is replaced by the ellipticity condition described above.

Pricing Derivatives with PDEs
Well posed problems vs ill conditioned problems

- Parabolic equations are very special in that they posses the regularising effect.
- This is in essence related to the irreversibility of time understood as the increase of entropy in all known physical processes.
- As a result it is impossible to demand that at time $t = 1$ the heat equation starting at $t = 0$ results in a discontinuous function such as a step function $u(x, 1) = 1_{[0,\infty)}(x)$. So the PDE problem on $(x, t) \in \mathbb{R} \times [0, 1]$:

  $$\begin{aligned} u_t &= u_{xx} \\ u(x,1) &= 1_{[0,\infty)}(x) \end{aligned}$$

  where $1_A(x)$ is the function that is 1 if $x \in A$ and 0 otherwise, has no solution. This is called an *ill conditioned* problem.
- The example above illustrates the fact that adding final solutions to the heat equation is an ill conditioned problem.

Pricing Derivatives with PDEs

Well posed problems vs ill conditioned problems

- A well conditioned problem is a problem that
  1. has a unique solution,
  2. the solution depends continuously on the initial data.

  It can be seen that adding initial conditions to a parabolic PDE is a well conditioned problem.

Pricing Derivatives with PDEs

Time reversed parabolid PDEs

Are pricing PDEs parabolic?

- The BS PDE with time dependent parameters is:

$$\frac{\partial f}{\partial t} = -\frac{1}{2}\sigma(t)^2\frac{\partial^2 f}{\partial s^2} + r(t)f - \left(r(t) - \frac{\sigma(t)^2}{2}\right)\frac{\partial f}{\partial s}.$$

- This is not parabolic as the coefficient of $\partial^2 f/\partial s^2$ is not positive!
- This is what is called a *"time reversed"* parabolic PDE (sometimes we do employ the attribute parabolic to simplify)
- The regularising effect described above now takes place but with the reversed time $t^* = -t$ or $t^* = T - t$. Note that $\partial f/\partial t = -\partial f/\partial t^*$.
- So for a time reversed parabolic PDE an initial condition is an ill defined problem whereas final conditions are well posed.
- A time reversed parabolic PDE regularises solutions as we decrease time. So if $f(s, T) = 1_{(0,\infty)}(s)$ then $f(s, t)$ for $t < T$ will be a smoothed version of this step function.

Numerical Resolution of PDEs

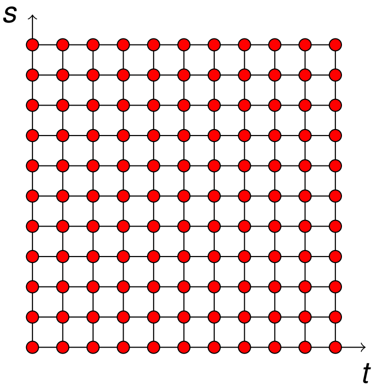
The Finite Difference Grid method

- There are several methods to approximate the solution of PDEs.
- We are going to study the Finite Difference Grid method.
- This is based on two basic ideas:
  1. Approximate derivatives by finite differences. E.g. $f'(x) \approx (f(x+\varepsilon) - f(x))/\varepsilon$ for small ε
  2. Replace the continuous domain $(s, t) \in \mathbb{R} \times [0, T]$ by a finite grid of points.

Discretization

A PDE is the problem of finding a function. As such it depends on values at infinitely many points.

In order to make this problem numerically tractable we need to make it finite. To do so we will discretise space and time by only considering spot and time at a finite set of values



Discretization

- To do this we divide time from now, $t = 0$, to expiry, $t = T$, in N intervals of size $\Delta t = T/N$:

$$t_0 = 0, t_1 = \Delta t, t_2 = 2\Delta t, \ldots, t_N = N\Delta t = T$$

- Likewise we will pick an upper and lower log-spot levels s_l and s_u and divide this interval in M intervals of size $\Delta s = (s_U - s_L)/M$:

$$s_0 = s_L, s_1 = s_L + \Delta s, \ldots, s_M = s_U$$

- In general it is not an easy problem to choose s_L and s_U except in the case that they feature in the derivatives contract as happens for instance in a Double-No-Touch.
- To simplify the explanation we will assume that we are dealing with a contract with a lower and upper barrier at B_L and B_U and then set the boundaries log grid to $s_L = \ln B_L$ and $s_U = \ln B_U$.

Discretization

- We will devise a scheme to approximate the value of the solution f at the node points $f(S_i, t_j)$ by a number $f_{i,j}$
- In order to determine these approximations $f_{i,j}$ we will need to convert our PDE to an equation in these variables.
- This is done by replacing the derivatives in the PDE by approximations involving the function value at the node points

Approximating derivatives
Finite differences

The derivative of a function f at x is formally defined as $\lim_{h\to 0}(f(x+h)-f(x))/h$. Therefore a sensible approximation to the derivative is

$$f'(x) \approx \frac{f(x+\Delta x)-f(x)}{\Delta x}$$

where Δx is sufficiently small.
It would be good if we could quantify how good/bad this approximation is.
To do this we use the Taylor expansion

$$f(x+\Delta x) = f(x) + f'(x)\Delta x + \frac{1}{2}f''(x)\Delta x^2 + \frac{1}{3!}f'''(x)\Delta x^3 + \cdots$$

Approximating derivatives

One sided derivatives

We can see that the highest order error term in the previous approximation is $f''(x)\Delta x/2$

$$f'(x) = \frac{f(x+\Delta x) - f(x)}{\Delta x} - \frac{1}{2}f''(x)\Delta x - \frac{1}{3!}f'''(x)\Delta x^2 + \cdots$$

Can we improve on this? Yes we can.
Just average the formula above with Δx and $-\Delta x$

Approximating derivatives

Central differences

Just average the formula above with Δx and $-\Delta x$

Forward Difference approximation to derivative

$$f'(x) = \frac{f(x+\Delta x) - f(x)}{\Delta x} - \frac{1}{2} f''(x) \Delta x + \cdots$$

Backward Difference approximation to derivative

$$f'(x) = \frac{f(x-\Delta x) - f(x)}{-\Delta x} + \frac{1}{2} f''(x) \Delta x + \cdots$$

and we obtain

Central Difference approximation to derivative

$$f'(x) = \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} - \frac{1}{3!} f'''(x) \Delta x^2 + \cdots$$

Approximating derivatives

Second derivatives

If instead of averaging the forward and backward approximations, we take the difference we get

Approximation to second derivative

$$f''(x) = \frac{f(x+\Delta x) - 2f(x) + f(x-\Delta x)}{\Delta x^2} - \frac{2}{4!} f^{(iv)}(x)\Delta x^2 + \cdots$$

Approximating derivatives
Discretization of the PDE

The idea now is to replace our PDE, say the BS PDE:

$$\frac{\partial f}{\partial t} + \frac{1}{2}\sigma^2 \frac{\partial^2 f}{\partial s^2} = rf - \left(r - \frac{1}{2}\sigma^2 \right) \frac{\partial f}{\partial s}$$

by an equation on the node points (s_i, t_j) using approximate derivatives.
This will lead to equations in the quantities $f_{i,j}$ that happen to be linear and tractable. The resulting scheme is similar to pricing on a tree but with some important improvements.

Approximating derivatives
Discretization of the PDE

We will substitute

- $\frac{\partial f}{\partial s}(s_i, t_j)$ by the central difference

$$\frac{f_{i+1,j} - f_{i-1,j}}{2\Delta s}$$

- $\frac{\partial^2 f}{\partial s^2}(s_i, t_j)$ by

$$\frac{f_{i+1,j} - 2f_{i,j} + f_{i-1,j}}{\Delta s^2}$$

- $\frac{\partial f}{\partial t}(s_i, t_j)$ by either of the one sided derivatives

$$\frac{f_{i,j+1} - f_{i,j}}{\Delta t} \text{ (forward in time)}$$

or

$$\frac{f_{i,j} - f_{i,j-1}}{\Delta t} \text{ (backward in time)}$$

Finite Differences for the BS PDE
Backwards in time

The finite difference approximation of the Black Scholes PDE using backward derivative in time is:

$$\frac{f_{i,j} - f_{i,j-1}}{\Delta t} + \frac{1}{2}\sigma^2 \frac{f_{i+1,j} - 2f_{i,j} + f_{i-1,j}}{\Delta s^2} = rf_{i,j} - \left(r - \frac{1}{2}\sigma^2\right) \frac{f_{i+1,j} - f_{i-1,j}}{2\Delta s}$$

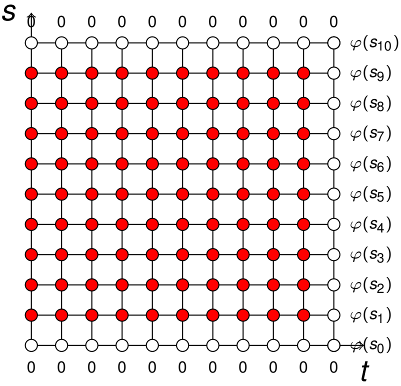
This is a linear equation in all the quantities $f_{i,j}$.

We should be analyse how tractable this linear equation is. If we have N time steps, and M spot steps, there is a total of MN unknown variables $f_{i,j}$. Resolving a general $n \times n$ linear system costs around $O(n^3)$ operations, which in this case is $O(N^3 M^3)$, this number could easily get out of hand, eg. $N = M = 1{,}000$ then $N^3 M^3 = 10^{18}$.

We want to explore whether these linear systems are easier to solve than the general $NM \times NM$ system.

Finite Differences for the BS PDE
Backwards in time

Note that on our grid, the value on the nodes on the last time step (S_i, t_N), is just the known payout $f_{i,N} = \varphi(s_i)$. We are also assuming the values in the upper and lower spot boundaries is zero $f_{0,j} = f_{M,j} = 0$.



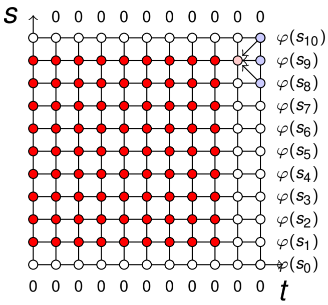
Finite Differences for the BS PDE

Backwards in time

Next note that if we make $j = N$ in the backwards equation we get

$$\frac{f_{i,N} - f_{i,N-1}}{\Delta t} + \frac{1}{2}\sigma^2 \frac{f_{i+1,N} - 2\, f_{i,N} + f_{i-1,N}}{\Delta s^2} = r\, f_{i,N} - \left(r - \frac{1}{2}\sigma^2\right) \frac{f_{i+1,N} - f_{i-1,N}}{2\Delta s}$$

Where all the final nodes $f_{i,N}$ are known, and the only unknown is $f_{i,N-1}$. The unknown is easily calculated as a linear combination of $f_{i-1,N}$, $f_{i,N}$ and $f_{i-1,N}$.

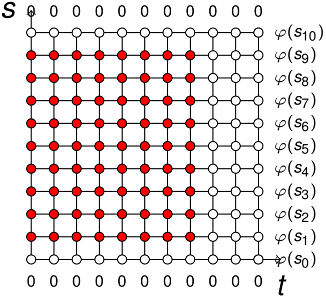


Finite Differences for the BS PDE
The explicit method

Having calculated all the nodes at time $t = t_{N-1}$ we write the backwards equation with $j = N - 1$

$$\frac{f_{i,N-1} - f_{i,N-2}}{\Delta t} + \frac{1}{2}\sigma^2 \frac{f_{i+1,N-1} - 2f_{i,N-1} + f_{i-1,N-1}}{\Delta s^2} = rf_{i,N-1} - \left(r - \frac{1}{2}\sigma^2\right) \frac{f_{i+1,N-1} - f_{i-1,N-1}}{2\Delta s}$$

From this we can calculate very easily all the $f_{i,N-2}$.



Finite Differences for the BS PDE
The explicit method

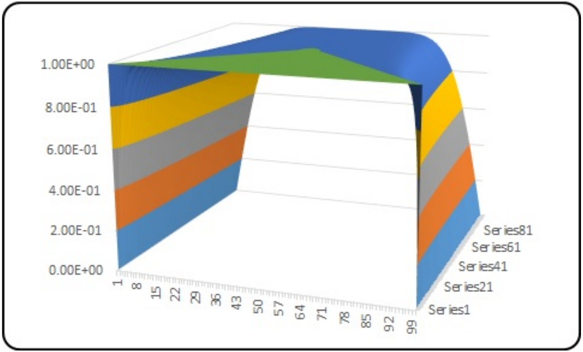
Applying this repeatedly we can calculate all the $f_{i,j}$ without the need to solve any huge *NM* linear system.
This is called the *explicit method*.

$$\begin{aligned} f_{i,j-1} &= f_{i-1,j}\left(\frac{1}{2}\sigma^2\frac{\Delta t}{\Delta s^2}-\frac{1}{2}\left(r-\frac{1}{2}\sigma^2\right)\frac{\Delta t}{\Delta s}\right)+f_{i,j}\left(1-\sigma^2\frac{\Delta t}{\Delta s^2}-r\Delta t\right) \\ &+ f_{i+1,j}\left(\frac{1}{2}\sigma^2\frac{\Delta t}{\Delta s^2}+\frac{1}{2}\left(r-\frac{1}{2}\sigma^2\right)\frac{\Delta t}{\Delta s}\right) \end{aligned}$$

Note the similarity with the trinomial tree method where we likewise move backwards in time and where the value at a node is a linear combination of the values at three nodes in the next time step.

Finite Differences for the BS PDE
Examples

If we price a 6M DNT on a 100 × 100 PDE grid we get something like:

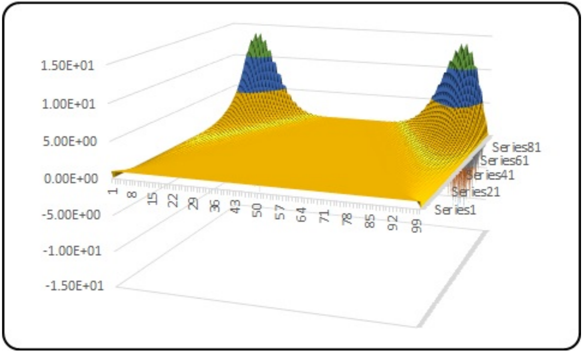


which looks reasonable.
The value of the range at expiry is “diffusing” as we move back in time from expiry.

Finite Differences for the BS PDE

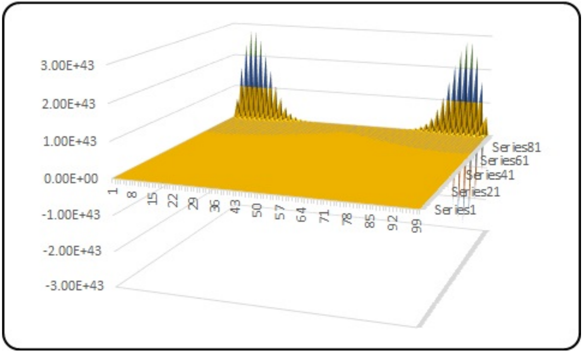
Examples

However if we increase time to expiry a bit to $T = 0.52$ on the same 100×100 grid we get:



which looks wrong.

And increasing to $T = 1$ produces:



which looks completely wrong.
Is there a bug?

Finite Differences for the BS PDE

Implementation

In connection with bug/hunting. When writing complicated functionality such as a PDE solver the recommended way of proceeding is:

1. Write a simple version on a very simple framework. For example calculate a PDE grid with $N = 2$ and $M = 3$ both by hand and with Excel. Compare and debug until it is 100% correct. The purpose of this is to produce a simple toy example that displays all numbers involved where we can verify by hand everything.
2. Write a simple version in a high level programming language that allows arbitrary N and M. Validate all intermediate calculations against the prototype above. Debug until all intermediate numbers are identical. Writing in high level language means that you can focus on the algorithm and do not have to worry at this stage of delicate memory allocation problems.

Finite Differences for the BS PDE

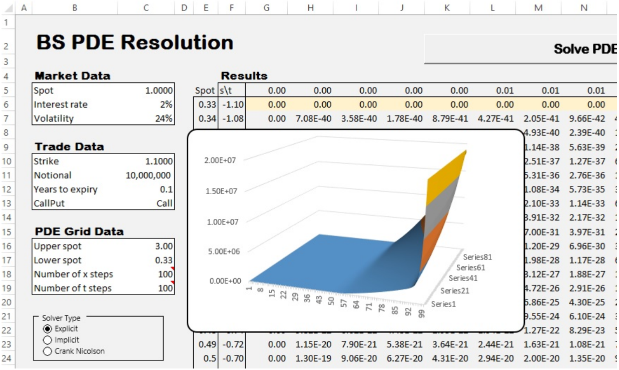
Implementation

3. Move to low level language. Make it temporarily return all intermediate calculations so that you can marry these numbers with the ones you get with the high level implementation. At this phase you will be basically focussing on performance, memory, bugs unrelated to the logic of the algorithm etc.

This way of working requires more work. But often this is less than the debugging time involved in resolving difficult problems in a low level implementation where you do not know whether the problem lies in the algorithm, a bug, a memory problem etc.

Finite Differences for the BS PDE
Examples

The following graph displays the graph of a vanilla option where we have set sensible (but unreal) boundary conditions.



BS PDE Resolution

Solve PDE

Market Data

Spot	1.0000
Interest rate	2%
Volatility	24%

Trade Data

Strike	1.1000
Notional	10,000,000
Years to expiry	0.1
CallPut	Call

PDE Grid Data

Upper spot	3.00
Lower spot	0.33
Number of x steps	100
Number of t steps	100

Solver Type
● Explicit
○ Implicit
○ Crank Nicolson

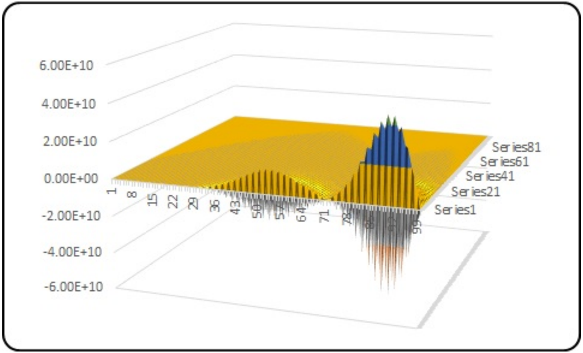
Results

Spot	s\t	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.01
0.33	-1.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.34	-1.08	0.00	7.08E-40	3.58E-40	1.78E-40	8.79E-41	4.27E-41	2.05E-41	9.66E-42
								4.93E-40	2.39E-40
								1.14E-38	5.63E-39
								2.51E-37	1.27E-37
								5.31E-36	2.76E-36
								1.08E-34	5.73E-35
								2.10E-33	1.14E-33
								3.91E-32	2.17E-32
								7.00E-31	3.97E-31
								1.20E-29	6.96E-30
								1.98E-28	1.17E-28
								3.12E-27	1.88E-27
								4.72E-26	2.91E-26
								5.86E-25	4.30E-25
								9.55E-24	6.10E-24
								1.27E-22	8.29E-23
0.49	-0.72	0.00	1.15E-20	7.90E-21	5.38E-21	3.64E-21	2.44E-21	1.63E-21	1.08E-21
0.5	-0.70	0.00	1.30E-19	9.06E-20	6.27E-20	4.31E-20	2.94E-20	2.00E-20	1.35E-20

As time goes back, the terminal payoff, $Payoff(S) = Notional \cdot (S - K)^+$, diffuses and becomes more smooth.

Finite Differences for the BS PDE
Examples

If in the same implementation we increase the time to expiry we get graphs like



Why is this happening?

Finite Differences for the BS PDE

Stability of the explicit method

Numerical methods can sometimes become unstable. In the examples above we see that as time to expiry increases the explicit method becomes unstable. Why is this happening?
Time to expiry becoming larger means that Δt is becoming larger. What effect does this have on the explicit method?

$$
\begin{aligned}
f_{i,j-1} &= f_{i-1,j}\left(\frac{\sigma^2}{2}\frac{\Delta t}{\Delta s^2} - \frac{1}{2}\left(r - \frac{1}{2}\sigma^2\right)\frac{\Delta t}{\Delta s}\right) + f_{i,j}\left(1 - \sigma^2\frac{\Delta t}{\Delta s^2} - r\Delta t\right) \\
&+ f_{i+1,j}\left(\frac{1}{2}\sigma^2\frac{\Delta t}{\Delta s^2} + \frac{1}{2}\left(r - \frac{1}{2}\sigma^2\right)\frac{\Delta t}{\Delta s}\right)
\end{aligned}
$$

This equation can be interpreted as making $f_{i,j-1}$ equal to an average of $f_{i-1,j}$, $f_{i,j}$, and $f_{i+1,j}$ using certain weights. We would expect these weights, as in a tree, to be in $[0,1]$ so that effectively we are averaging.

Finite Differences for the BS PDE

Stability of the explicit method

- As Δs and Δt become smaller the term in the equation above that could explode is $\Delta t/\Delta x^2$.
- For example if $\Delta s = \Delta t = 10^{-3}$ then $\Delta t/\Delta x^2 = 1{,}000$ and this large number causes the weights above to be very large positive and very large and negative.
- This will cause instabilities because small values of $f_{i,j}$ can amplify to large values of $f_{i,j-1}$ and grow out of control.
- This is the phenomenon displayed in the previous slides.
- So in order for the explicit method to converge we must have that $\Delta t/\Delta s^2$ is small. This will be quantified in a later lesson.
- This requires M to become larger which costs potentially expensive CPU power and processing time.

Finite Differences for the BS PDE

The implicit method

So to summarize:

+ Explicit method is easy.
- The explicit method is unstable unless Δt is very small. This means that the number of time steps has to be large which will increase the computational cost.

Finite Differences for the BS PDE
The implicit method

As usual we ask the question: Can we do better than this?
The answer is: Yes we can.
If instead of using backwards approximation to the time derivative, we use the forward approximation:

$$\frac{f_{i,j+1}-f_{i,j}}{\Delta t}+\frac{1}{2}\sigma^2\frac{f_{i+1,j}-2f_{i,j}+f_{i-1,j}}{\Delta s^2}=rf_{i,j}-\left(r-\frac{1}{2}\sigma^2\right)\frac{f_{i+1,j}-f_{i-1,j}}{2\Delta s}$$

we get a method with slightly different properties.

Finite Differences for the BS PDE

The implicit method

In particular if we take $j = N - 1$ we get again an equation linking the values at the last time step ($f_{i,N}$) with the values on the previous time step ($f_{i,N-1}$):

$$\begin{aligned} \frac{f_{i,N} - f_{i,N-1}}{\Delta t} \quad + \quad & \frac{1}{2}\sigma^2 \frac{f_{i+1,N-1} - 2\, f_{i,N-1} + f_{i-1,N-1}}{\Delta s^2} \\ = \quad & r\, f_{i,N-1} - \left(r - \frac{1}{2}\sigma^2\right) \frac{f_{i+1,N-1} - f_{i-1,N-1}}{2\Delta s} \end{aligned}$$

Where the known values are the ones at the last time $j = N$, $f_{i,N}$, and the unknowns are the values on the time before expiry $j = N - 1$, $f_{i,N-1}$.

This is a set of linear equations if we let $i = 1, \dots M - 1$ but it is not as easy as in the explicit method.

This method is called implicit.

Finite Differences for the BS PDE

The implicit method

We can rewrite the equations above like

$$\left(-\frac{\sigma^2\Delta t}{2\Delta s^2}+\left(r-\frac{\sigma^2}{2}\right)\frac{\Delta t}{2\Delta s}\right)f_{i+1,N-1}+\left(1+r\Delta t+\frac{\sigma^2\Delta t}{\Delta s^2}\right)f_{i,N-1}+\left(-\frac{\sigma^2\Delta t}{2\Delta s^2}-\left(r-\frac{\sigma^2}{2}\right)\frac{\Delta t}{2\Delta s}\right)f_{i-1,N-1}=f_{i,N}$$

where we have $M-1$ unknowns $f_{1,N-1},\ldots f_{M-1,N-1}$ (recall that the boundary conditions imply $f_{0,N-1}=0$ and $f_{M,N-1}=0$) and the $M-1$ equations as above with $i=1\ldots M-1$.
These equations are called tridiagonal and are much easier to resolve than the standard Gauss resolution of a full $(M-1)\times(M-1)$ system (c.f. the Thomas algorithm in the lesson on Splines)

Finite Differences for the BS PDE
The implicit method

If we write

$$
\begin{aligned}
a &= -\frac{\sigma^2 \Delta t}{2\Delta s^2} + \left(r - \frac{1}{2}\sigma^2\right)\frac{\Delta t}{\Delta s}, \\
b &= 1 + r\Delta t + \frac{\sigma^2 \Delta t}{\Delta s^2}, \\
c &= -\frac{\sigma^2 \Delta t}{2\Delta s^2} - \left(r - \frac{1}{2}\sigma^2\right)\frac{\Delta t}{\Delta s}
\end{aligned}
$$

then the system to resolve is

$$
\begin{pmatrix}
b & c & \cdots & & \\
a & b & c & & \\
\vdots & & \ddots & & \\
 & & a & b & c \\
 & & & a & b
\end{pmatrix}
\begin{pmatrix}
f_{1,N-1} \\
f_{2,N-1} \\
\vdots \\
f_{M-1,N-1}
\end{pmatrix}
=
\begin{pmatrix}
f_{1,N} \\
f_{2,N} \\
\vdots \\
f_{M-1,N}
\end{pmatrix}
$$

Finite Differences for the BS PDE

Stability analysis: time-slices as vectors

- In order to analyse how our approximations evolve it is useful to view all approximations at a given time slice *j* as a vector

$$f_j = \begin{pmatrix} f_{1,j} \\ f_{2,j} \\ \vdots \\ f_{M-1,j} \end{pmatrix}.$$

- Then the implicit method above can be described as

$$Af_{j-1} = f_j$$

  where *A* is the tridiagonal matrix of the system in the previous slide.
- If we know f_N then the previous time-slices can be calculated as $f_{N-1} = A^{-1} f_N$, $f_{N-2} = A^{-1} f_{N-1} = A^{-2} f_N$, $f_{N-3} = A^{-3} f_N, \ldots$
- The method will be stable if as we calculate powers of the matrix A^{-1} we get numbers that do not go to infinity.

Finite Differences for the BS PDE

Stability analysis: time-slices as vectors

- The explicit method can also be described in a similar fashion.
- If we set

$$a' = \frac{\sigma^2}{2}\frac{\Delta t}{\Delta s^2} - \left(r - \frac{1}{2}\sigma^2\right)\frac{\Delta t}{2\Delta s} \qquad c' = \frac{\sigma^2}{2}\frac{\Delta t}{\Delta s^2} + \left(r - \frac{1}{2}\sigma^2\right)\frac{\Delta t}{2\Delta s}$$

$$b' = 1 - \sigma^2\frac{\Delta t}{\Delta s^2} - r\Delta t$$

the explicit equations are

$$f_{i,j-1} = a' f_{i-1,j} + b' f_{i,j} + c' f_{i+1,j}, \quad i = 1, \ldots, M-1.$$

- This can be written in matrix notation as $f_{j-1} = Bf_j$ if B is the tridiagonal matrix with b' on the diagonal. a' below the diagonal, and c' above the diagonal.
- The explicit method will be stable provided that powers of B do not grow indefinitely.

Finite Differences for the BS PDE

Stability analysis: definition of stability

- In order to establish whether the implicit method is stable we need to see whether A^{-j} grows too much as j increases.
- For the implicit method we need to see whether B^j grows as j increases. We empirically known that the explicit method can become unstable if Δt is not small enough.
- How do we define the size of a matrix?

Finite Differences for the BS PDE

Stability analysis: definition of stability

- A way to measure the size of a matrix is to see how long it can make vectors.
- In general the vector Mv is not directly comparable to the vector v because it points in a different direction.
- But sometimes Mv is just a longer or shorter version of v: $Mv = \lambda v$.
- We call such vectors *eigenvectors* and the amplification factor λ an eigenvalue.
- In general a matrix does not need to have eigenvalues or eigenvectors.
- The matrices we will have plenty of eigenvectors and eigenvalues.

Finite Differences for the BS PDE

Stability analysis: definition of stability

- Given a $n \times n$ matrix M, if there exist a basis of vectors e_1,...,e_n such that $Me_i = \lambda_i e_i$ then M is said to be diagonalizable.
- If a matrix M is diagonalizable and we have that all eigenvalues are smaller or equal than 1 in absolute value (i.e. $|\lambda_i| \leq 1$ for $i = 1, \dots n$) then M is said to be stable. For any vector f we have that the size of the vector $M^j f$ is smaller or equal than the size of f for all $j \in \mathbb{N}$.
- If a matrix M is diagonalizable and we have that at least one eigenvalues is larger than 1 in absolute value (i.e. there is a i such that $|\lambda_i| > 1$) then M is said to be unstable. Then there exist vectors f such that $M^j f$ increases indefinitely in size as j increases.
- Note that the eigenvalues of M^{-1} are the inverses of the eigenvalues of M. Therefore if M is stable and $M \neq Id$ then M^{-1} is unstable.

Finite Differences for the BS PDE

Stability analysis

- Calculating eigenvalues is a complicated business.
- Even for tridiagonal matrices there is no analytic formula.
- However if the matrix is tridiagonal with constant diagonals then there is an analytic formula. Such matrices are called Toepliz. A Toepliz matrix is a matrix $A = (a_{ij})$ with $a_{ij} = a_{kl}$ if $i - j = k - l$ (i.e. constant across diagonals)
- The eigenvalues of a $n \times n$ tridiagonal matrix with b on the diagonal, c above, and a below are

$$\lambda_i = b + 2\sqrt{ac}\cos\left(\frac{i\pi}{n+1}\right) \text{ for } i = 1, \ldots n.$$

Finite Differences for the BS PDE

Stability analysis: the implicit method

So what are the eigenvalues of the matrix A in the implicit method? If we substitute a, b and c in the equation in the previous slide, for $i = 1, \dots n$ we get:

$$\begin{aligned}
\lambda_i &= b + 2\sqrt{ac}\cos\left(\frac{i\pi}{n+1}\right) = \\
&= 1 + \sigma^2\frac{\Delta t}{\Delta s^2} + r\Delta t + 2\sqrt{\left(\frac{\sigma^2}{2}\frac{\Delta t}{\Delta s^2}\right)^2 - \left(r - \frac{\sigma^2}{2}\right)^2\left(\frac{\Delta t}{2\Delta s}\right)^2}\cos\left(\frac{i\pi}{n+1}\right) \\
&> 1 + \sigma^2\frac{\Delta t}{\Delta s^2} - \sqrt{\left(\sigma^2\frac{\Delta t}{\Delta s^2}\right)^2 - \left(r - \frac{\sigma^2}{2}\right)^2\left(\frac{\Delta t}{\Delta s}\right)^2} \\
&> 1.
\end{aligned}$$

(in the first inequality we have assume $r\Delta t > 0$ and replaced it by zero, and replace $\cos(\cdot)$ by its smallest value -1.)

As the eigenvalues of A^{-1} are λ_i^{-1}, by the equation above $\lambda_i^{-1} \in (0, 1)$ and therefore the implicit method is stable.

Finite Differences for the BS PDE
Stability analysis: the explicit method

What are the eigenvalues of the matrix B in the explicit method?
If we substitute a', b' and c' in the equation in the previous slide, for $i = 1, \ldots n$ we get:

$$\begin{aligned}\lambda_i &= b' + 2\sqrt{a'c'}\cos\left(\frac{i\pi}{n+1}\right) = \\ &= 1 - \sigma^2\frac{\Delta t}{\Delta s^2} - r\Delta t + 2\sqrt{\left(\frac{\sigma^2}{2}\frac{\Delta t}{\Delta s^2}\right)^2 - \left(r - \frac{\sigma^2}{2}\right)\left(\frac{\Delta t}{2\Delta s}\right)^2}\cos\left(\frac{i\pi}{n+1}\right).\end{aligned}$$

As it is harder to analyse when this expression is smaller in absolute value than 1, typically one approximates it by assuming that the terms with Δt and $\Delta t/\Delta s$ are negligible compared to the terms with $\Delta t/\Delta s^2$. Then

$$\lambda_i \approx 1 - \sigma^2\frac{\Delta t}{\Delta s^2} + 2\sqrt{\left(\frac{\sigma^2}{2}\frac{\Delta t}{\Delta s^2}\right)^2}\cos\left(\frac{i\pi}{n+1}\right) = 1 - \left(1 - \cos\left(\frac{i\pi}{n+1}\right)\right)\sigma^2\frac{\Delta t}{\Delta s^2}.$$

Stability analysis: the explicit method

- We can simplify this by using the identity $\sin^2(\theta) = (1 - \cos(2\theta))/2$:

$$\lambda_i \quad \approx \quad 1 - \left(1 - \cos\left(\frac{i\pi}{n+1}\right)\right)\sigma^2\frac{\Delta t}{\Delta s^2} = 1 - 2\sin^2\left(\frac{i\pi}{2(n+1)}\right)\sigma^2\frac{\Delta t}{\Delta s^2}$$

- This formula tells us that the (approximated) eigenvalues are always smaller than $+1$.
- The explicit method can be unstable if they are below -1.
  - If $\sigma^2\frac{\Delta t}{\Delta s^2}$ is small then the term with the sine will be small so that the eigenvalue will not be below -1.
  - If $\sigma^2\frac{\Delta t}{\Delta s^2}$ is large then the term with the sine can become very large and negative and the eigenvalue will be smaller than -1.

Finite Differences for the BS PDE

Stability analysis: the explicit method

Homework:

- Check that the smallest eigenvalue is $\lambda_n = 1 - 2\sin^2\left(\frac{n\pi}{2(n+1)}\right)\sigma^2\frac{\Delta t}{\Delta s^2}$.
- Show that the explicit method is stable if and only if $\sigma^2\frac{\Delta t}{\Delta s^2} < 1/\sin^2(n\pi/(2(n+1)))$.
- Show that if n is large this criterion can be approximated by $\sigma^2\frac{\Delta t}{\Delta s^2} < 1$

Finite Differences for the BS PDE

Stability analysis: Summary

- The explicit method can be written as $f_j = Bf_{j+1}$. Where f_j is the vector of solutions at time j and B is a certain Toepliz tridiagonal matrix.
- The implicit method can be written as $f_j = A^{-1} f_{j+1}$. Where A is a certain Toepliz tridiagonal matrix.
- The implicit method is stable if the eigenvalues of A^{-1} are smaller in absolute value than 1. This is proved to be always the case: the implicit method is *unconditionally* stable.
- The explicit method is stable if the eigenvalues of B are smaller in absolute value than 1. Modulo some approximations this will be the case if and only if $\Delta t < \Delta s^2/\sigma^2$. The explicit method is *conditionally stable*.
- Therefore for the explicit method to work we need Δt to be small which means we must use many time steps. This can compromise performance.

Finite Differences for the BS PDE

Stability analysis: discrete Fourier analysis

- The eigenvalue analysis presented in the previous slides is often phrased in the language of discrete Fourier analysis.
- If we happen to know the eigenvectors of a matrix M: $e_1, \dots e_n$ (this means $Me_i = \lambda_i e_i$ with an eigenvalue λ_i) then successive applications of the matrix M on a eigenvector e_i result in successive amplifications

$$\begin{aligned} M^2 e_i &= MMe_i = M\lambda_i e_i = \lambda_i Me_i = \lambda_i^2 e_i && (5) \\ M^3 e_i &= MMMe_i = \cdots = \lambda_i^3 e_i && (6) \\ &\vdots && (7) \end{aligned}$$

- In our context λ_i are also called *amplification factors* because they amplify (or decrease if $|\lambda_i| < 1$) the vectors e_i.
- The vectors e_i are sometimes called *Fourier modes* or *fundamental harmonics*.

Finite Differences for the BS PDE

Stability analysis: discrete Fourier analysis

- The idea is that our payoff vector can be written as a linear combination of Fourier modes

$$f_N = a_1 e_1 + \cdots + a_n e_n$$

where e_i are the eigenvectors of say the explicit method matrix B.

- To move back a time slice, say in the explicit method we just need to apply the matrix B

$$\begin{aligned} f_{N-1} &= Bf_N = B\left(a_1 e_1 + \cdots + a_n e_n\right) = a_1 Be_1 + \cdots + a_n Be_n \\ &= a_1 \lambda_1 e_1 + \cdots + a_n \lambda_n e_n, \\ f_{N-2} &= a_1 \lambda_1^2 e_1 + \cdots + a_n \lambda_n^2 e_n, \\ f_{N-3} &= a_1 \lambda_1^3 e_1 + \cdots + a_n \lambda_n^3 e_n, \\ &\vdots \end{aligned}$$

- This is mostly of theoretical interest.

Finite Differences for the BS PDE

Stability analysis: discrete Fourier analysis

- For example, if the method is unstable the eigenvalue largest in absolute value can be seen to be:

$$\lambda_n \approx 1 - 2\sin^2\left(\frac{n\pi}{2(n+1)}\right)\sigma^2\frac{\Delta t}{\Delta s^2}.$$

- If for our particular payoff a_n is close to zero then we should be more or less immune to this source of instability because λ_n does not affect the equation for f_{N-1}, f_{N-1} ,... in the previous slide.
- It can be seen that the eigenvector e_n is a vector that changes sign at each component: it is the discrete analogue of a very high frequency sine curve.
- Philosophically if our payoff is nice and smooth a_n should be small as the graph of f_N is not very oscillating. The continuous version of this statement is called the Riemann-Lebesgue Lemma.
- However a payoff that is discontinuous might have a larger a_n and be more exposed to instabilities.

Finite Differences for the BS PDE

Stability analysis: discrete Fourier analysis

Riemann-Lebesgue theorem

If $f : [0, 2\pi] \longrightarrow \mathbb{R}$ is an integrable function, $f \in L^1[0, 2\pi]$, then the Fourier coefficients

$$a_n = \int_0^{2\pi} f(x)\mathrm{e}^{nxi}\mathrm{d}x$$

converge to zero $a_n \to 0$.

By using integration by parts it is easy to see that the n-th Fourier coefficient of the k-th derivative $f^{(k)}$ is that of f multiplied by $(in)^k$. One can deduce the following result:

If $f : [0, 2\pi] \longrightarrow \mathbb{R}$ is k times differentiable, $f \in C^k[0, 2\pi]$, then the Fourier coefficients converge to zero faster than $1/n^k$:

$$\lim_{n\to\infty} n^k a_n = 0.$$

Finite Differences for the BS PDE

Stability analysis: discrete Fourier analysis

There are formulæ for the eigenvectors of a general Toepliz tridiagonal matrix:

$$M = \begin{pmatrix} b & c & \cdots & & & \\ a & b & c & & & \\ \vdots & & \ddots & & & \\ & & & a & b & c \\ & & & & a & b \end{pmatrix}$$

The eigenvector with eigenvalue $\lambda_i = b + \sqrt{ac}\cos\left(\frac{i\pi}{n+1}\right)$ (i from 1 to n) is:

$$e_i = \left(\left(\frac{a}{c}\right)^{1/2}\sin\left(\frac{i\pi}{n+1}\right), \cdots, \left(\frac{a}{c}\right)^{n/2}\sin\left(\frac{ni\pi}{n+1}\right),\right)$$

Homework: Check that $Me_i = \lambda_i e_i$.

Homework: Draw a graph of the function $f_i(x) = \left(\frac{a}{c}\right)^{x/2}\sin\left(\frac{ix\pi}{n+1}\right)$ and plot on it the components of the vector e_i. Assume $a = c$.

Finite Differences for the BS PDE

Stability analysis: discrete Fourier analysis

- Writing the vector f_N as linear combinations of the vectors e_i in the previous slide is the discrete analogous to expressing a function as a sum of sine functions

$$f(x) = \sum_{i=0}^{\infty} \widehat{f}(n) \sin(nx)$$

this is the content of classical Fourier analysis.

- For this reason using expressions such as $f_n = \sum a_i e_i$ is often called *discrete Fourier analysis*.
- Note that in the classical case the functions $\sin(nx)$ are the eigenvalues of the laplacian Δ as $\Delta \sin(nx) = \frac{\partial}{\partial x^2} \sin(nx) = -n \sin(nx)$.
- The eigenvalues of the Laplacian are the source of a lot of mathematical research. C.f. "Can you hear the shape of a drum?" at `http://www.jstor.org/stable/2313748`.

Other topics

Numerical resolution of PDEs is a huge topic with thousands of methods and variations. It is used in most of applied mathematics (oil prospection, fluid dynamics, etc).

Some suggested topics for further exploration

1. Revisit analysis above in the simpler context of the heat equation.
2. Crank Nicolson
3. Alternate Direction Implicit (ADI)