Fun with Finite Difference

December 2022

Jesper Andreasen
Saxo Bank, Copenhagen

kwant.daddy@saxobank.com

Abstract

We summarize most of what you need to know about finite difference solution of one-dimensional partial differential equations in finance. We demonstrate the theoretical results by numerical experiments. The reader is invited and encouraged to replicate the experiments using the C++ code and spreadsheets in our GitHub repository.

Introduction

The …

One More Time for Prince Knud

Before we get jiggy with the numbers, we'll quickly derive the theta scheme for numerical solution of one dimensional partial differential equations.

Consider a state variable x that evolves according to

$$dx = \mu(t,x)dt + \sigma(t,x)dW \tag{1}$$

where W is a Brownian motion under the risk neutral measure. Assume that the interest rate is a function of (t,x) only, i.e. $r = r(t,x)$.

Using Ito's lemma it is straightforward to show that the expectation

$$f(t,x(t)) = E_t[e^{-\int_t^T r(u)du} f(T,x(T))] \tag{2}$$

is the solution to the backward partial differential equation

$$0 = \partial_t f + Af \quad , A = -r + \mu\partial_x + \frac{1}{2}\sigma^2\partial_{xx} \tag{3}$$

with $f(T,x)$ as the terminal boundary condition.

The partial differential equation (3) is turned into a finite difference equation in two steps.

First, we discretise the space $x_0 < x_1 < \ldots < x_{n-1}$ and introduce the first order spatial difference operators

$$\begin{aligned}
\delta_x^+ f(x_i) &= \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \\
\delta_x^- f(x_i) &= \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} \\
\delta_x f(x_i) &= \frac{x_{i+1} - x_i}{x_{i+1} - x_{i-1}}\delta_x^- f(x_i) + \frac{x_i - x_{i-1}}{x_{i+1} - x_{i-1}}\delta_x^+ f(x_i)
\end{aligned} \tag{4}$$

and second order spatial operator

$$\delta_{xx} f(x_i) = 2\frac{\delta_x^+ f(x_i) - \delta_x^- f(x_i)}{x_{i+1} - x_{i-1}} \tag{5}$$

Taylor expansion can be used to derive the order of the accuracy in (4) and (5).

These operators can be represented as rows of a tridiagonal matrix. Specifically,

$$\begin{aligned}
(\delta_x^+)_{i\cdot} &= \begin{bmatrix} 0 & \frac{-1}{x_{i+1}-x_i} & \frac{1}{x_{i+1}-x_i} \end{bmatrix} \qquad ,0 \le i < n-1 \\
(\delta_x^-)_{i\cdot} &= \begin{bmatrix} \frac{-1}{x_i - x_{i-1}} & \frac{1}{x_i - x_{i-1}} & 0 \end{bmatrix} \qquad ,0 < i \le n-1 \\
(\delta_x^-)_{0\cdot} &= (\delta_x^+)_{n-1,\cdot} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}
\end{aligned} \tag{6}$$

and

$$\begin{aligned}
(\delta_x)_{i\cdot} &= \frac{1}{x_{i+1}-x_{i-1}} \begin{bmatrix} -\frac{x_{i+1}-x_i}{x_i - x_{i-1}} & \frac{x_{i+1}-x_i}{x_i - x_{i-1}} - \frac{x_i - x_{i-1}}{x_{i+1}-x_i} & \frac{x_i - x_{i-1}}{x_{i+1}-x_i} \end{bmatrix} \qquad ,0 < i < n-1 \\
(\delta_x)_{0\cdot} &= (\delta_x^+)_{0\cdot} \quad ,(\delta_x)_{n-1,\cdot} = (\delta_x^-)_{n-1,\cdot}
\end{aligned} \tag{7}$$

and finally

$$\begin{aligned}
(\delta_{xx})_{i\cdot} &= \frac{2}{x_{i+1}-x_{i-1}} \cdot \begin{bmatrix} \frac{1}{x_i - x_{i-1}} & (\frac{-1}{x_i - x_{i-1}} + \frac{-1}{x_{i+1}-x_i}) & \frac{1}{x_{i+1}-x_i} \end{bmatrix} \qquad ,0 < i < n-1 \\
(\delta_{xx})_{0\cdot} &= (\delta_{xx})_{n-1,\cdot} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}
\end{aligned} \tag{8}$$

In the above we use a compact notation for tridiagonal matrices where we only list the three elements around the diagonal, in line with the band diagonal matrix representation used in Press et al (1988).

Depending on the application we will use either central differencing

$$\bar{A} = -r + \mu\delta_x + \frac{1}{2}\sigma^2\delta_{xx} \tag{9}$$

or dynamic up- and down-winding

$$\bar{A} = -r + \mu^+\delta_x^+ + \mu^-\delta_x^- + \frac{1}{2}\sigma^2\delta_{xx} \tag{10}$$

If we let $f(t) = (f(t,x_0),\ldots,f(t,x_{n-1}))'$, the PDE (3) can now replaced by a matrix equation

$$0 = \partial_t f + \bar{A}f \tag{11}$$

Over a discrete time step the solution to (11) is

$$f(t_h) = e^{\Delta t\bar{A}} f(t_{h+1}) = (\sum_{k=0}^{\infty} \frac{(\Delta t\bar{A})^k}{k!}) f(t_{h+1}) \qquad , \Delta t = t_{h+1} - t_h \tag{12}$$

Discretising (12) now leads to the theta scheme

$$\begin{aligned} f(t_{h+1/2}) &= [I + (1-\theta)\Delta t\bar{A}] f(t_{h+1}) \\ [I - \theta\Delta t\bar{A}] f(t_h) &= f(t_{h+1/2}) \end{aligned} \tag{13}$$

where I denotes the identity. The idea is now to solve (13) repeatedly

$$t_h \leftarrow t_{h+1/2} \leftarrow t_{h+1}$$

until we obtain the solution at time t_0. We will generally let m denote the number of time steps.

We generally let $\theta = 0$ denote the *explicit* scheme, $\theta = 1$ the fully *implicit* scheme and $\theta = 1/2$ the *Crank-Nicolson* scheme.

And now the other way around. Rearrange (13) and multiply it by the vectors $p(t_h), p(t_{h+1/2})$ and we get

$$\begin{aligned} p(t_{h+1/2})' f(t_{h+1/2}) &= p(t_{h+1/2})'[I + (1-\theta)\Delta t\bar{A}] f(t_{h+1}) \\ p(t_h)' f(t_h) &= p(t_h)'[I - \theta\Delta t\bar{A}]^{-1} f(t_{h+1/2}) \end{aligned} \tag{14}$$

If $p(t_h)$ is defined as the solution to the system

$$\begin{aligned} [I - \theta\Delta t\bar{A}]' p(t_{h+1/2}) &= p(t_h) \\ p(t_{h+1}) &= [I + (1-\theta)\Delta t\bar{A}]' p(t_{h+1/2}) \end{aligned} \tag{15}$$

Then p is the discrete Green's function for the backward linear system (13), i.e.

$$p(t_h)' f(t_h) = p(t_{h+1/2})' f(t_{h+1/2}) = p(t_{h+1})' f(t_{h+1}) \tag{16}$$

Hence, (15) defines a *forward* scheme for the discrete Green's function for the backward linear system (13). The scheme is solved by stepping forward in time

$$t_h \rightarrow t_{h+1/2} \rightarrow t_{h+1}$$

typically with the initial boundary condition $p(t_0, x_i) = 1_{x_i = x(t_0)}$.

Coding a Theta Solver

Coding a theta solver requires surprisingly little code. Actually, in terms of lines of code, almost as compact as code for the Black-Scholes formula and associated implied volatility calculator.

Consulting our GitHub repository you'll see that we use the following components to implement our theta solver:

1. kFiniteDifference::dx() and ::dxx(): construct the first and second order difference operators in compact matrix form (6-8).

2. kFd1d::calcAx(): constructs the matrix $I + \Delta t\bar{A}$ used in (9-10).

3. kMatrixAlgebra::banmul(): band diagonal matrix-vector multiplication (13a).

4. kMatrixAlgebra::tridag(): tridiagonal matrix-vector solution (13b).

5. kFd1d::rollBwd(): implements the backward roll by calling the above routines.

6. kMatrixAlgebra::transpose(): transposes a band diagonal matrix.

7. kFd1d::rollFwd(): implements the forward roll (15) by calling the above routines.

The routines banmul() and tridag() are pretty much straight copies of standard routines from Numerical Recipes by Press et al (1988).

Our code is compact and follows the mathematical matrix notation used in the previous section. No festival of little ants like in Figure 1. It's very well-shaped, IMUHO[1].

The first thing to check when you implement the theta solver is to reconcile the forward and backward solutions. The sheet 'duality' in the workbook fd1d shows that this indeed the case. The values in the two green fields match.

How Wide the Grid?

The rule of thumb is approximately ±5 standard deviations. Less than this will significantly reduce accuracy and more will just result in more exercise for the electrons in the computer without any benefit. The way to think about this rule is that finite difference solution is essentially a method of numerical integration, and as such you need to cover enough of the mass of the distribution but not necessarily more than that.

One modification to this rule is that for processes with significant mean-reversion, such as for example interest rate models, it appears to be a better rule to base the grid dimensioning on quadratic variation, i.e.

[1] In-My-Usual-Humble-Opinion.

$$x_{n-1} - x(t_0) \approx x(t_0) - x_0 \approx q \cdot [\int_0^{t_n} \sigma(u, x(t_0))^2 du]^{1/2} \quad (17)$$

with $q \approx 5$.

Another, modification is for very fat tailed distributions such as for example when stochastic volatility is involved, it appears that $q \approx 10$ is a better number.

In Table 2 we show the implied volatility of an option as function of the width of the grid. The sheet 'grid width' has been used to generate this figure. Note that as the grid is narrowed, probability mass builds up on the grid boundaries.

Wot PDE?

In many cases we transform the PDE to resemble the heat equation. You can for example choose to transform the constant parameter Black-Scholes PDE

$$0 = f_t - rf + \mu S f_S + \frac{1}{2}\sigma^2 S^2 f_{SS} \quad (18)$$

to

$$0 = f_t + \frac{1}{2} f_{xx} \qquad , S = S(0)e^{\mu t - \frac{1}{2}\sigma^2 t + \sigma x} \quad (19)$$

Why would you do so? If you don't, you can end up using a disproportional number of grid points for very extreme levels and almost none around where they actually matter. The problem increases with maturity of the finite difference grid and it is illustrated in Figure 3.

Transforming the PDE as in (17-18) increases the formal accuracy but it can have drawbacks, such as for example making it more difficult to control the martingale property of the spot. We discuss this in Andreasen (2022c). The alternative to transforming the PDE is to use a non-uniform grid spacing. For the case of (14-15), we could choose to solve the PDE on the grid points

$$S_i = S(0)e^{(i-n/2)\cdot\Delta x} \quad , i = 0, \ldots, n-1 \quad (20)$$

This will reduce the formal accuracy as the spacing will now be non-uniform. However, in practice, this is often a price worth paying.

Grid Point Placement

… is somewhat more tricky than one would think. Here we describe some of the approaches that can be used.

If the strike lands on the grid point the finite difference solution will tend to undervalue the option by a quantity of order $O(\Delta x^2)$. The bias disappears if the strike is exactly between two grid points. For digital options the bias is $O(\Delta x)$. This is not just an issue for the accuracy of the solution. Specifically, if the grid isn't aligned to the strikes there

will always be a noise of at least $O(\Delta x^2)$ in the system and this will particularly affect the Greeks.

So ideally, grid points should be placed so strike points are symmetrically surrounded by grid points.

An alternative is to smooth the payoff, i.e. to replace the payoff at level x_i by

$$\frac{2}{x_{i+1}-x_{i-1}}\int_{(x_{i-1}+x_i)/2}^{(x_i+x_{i+1})/2} f(x)dx \qquad (21)$$

This is a quite effective methodology and we illustrate this in figure 4 where we show the implied volatility of an option as function of the strike when the finite difference grid is kept fixed, for a non-smoothed and smoothed terminal payoff.

What about barriers? Barriers that are monitored at discrete times should be treated like digital options. I.e. either smooth the payoff or place the grid points symmetrically around the barrier. Continuously monitored barriers can be handled by placing the barrier level on the grid and enforcing absorption on the barrier by setting volatility and drift equal to zero on the barrier: $\mu = \sigma = 0$. Not doing so, will result in a dramatic accuracy loss of $O(\Delta t^{1/2})$.

What about the spot? Generally, we recommend placing the spot on the grid. If it isn't, the prices at the current time will have to be interpolated. Linear interpolation effectively corresponds to a one-step binomial tree, i.e. additional volatility will be injected into the solution. In the context of local volatility models we can correct for this but it may not always be possible. Higher order interpolation schemes like splines and polynomials will generally produce negative weights on specific points and this will in turn affect stability so we don't really like that route too much.

Stability

Conventionally, stability is analysed using von Neumann analysis. The idea is to consider eigen solutions to (13) of the form

$$g(t_h, x) = g^{-h}e^{ikx} \quad , i = \sqrt{-1}, k \in \mathbb{R}, g \in \mathbb{C} \qquad (22)$$

If all eigen solutions have the property that $|g| \leq 1$, then we say that the scheme is von Neumann stable. Strictly speaking, von Neumann stability is only sufficient to conclude that the scheme is convergent for the constant parameter case. However, there are, to my knowledge, no counter examples of it not being sufficient for non-constant parameters.

Von Neumann stability doesn't always imply that all transition probabilities are positive. But it means that as the number of time steps is increased the transition probabilities over multiple time steps converge to positive values, hence the numerical solution will ultimately converge to the continuous time and state solution as both time and spatial steps tend to zero $(\Delta t, \Delta x) \to (0,0)$.

For the explicit scheme, $\theta = 0$, von Neumann stability is ensured if and only the matrix $[I + \Delta t\bar{A}]$ has only non-negative elements. For the case of $r = \mu = 0$, this requires

$$\Delta t \leq \frac{\sigma^2}{\Delta x^2} \tag{23}$$

For the case when (23) holds with equality, the explicit scheme corresponds to a binomial tree. In the specific example in the sheet 'explicit' this occurs when the number of time steps is exactly 25. You can wary the number of time steps to see what happens.

The implicit scheme, $\theta = 1$, is always von Neumann stable and thereby convergent. For the drift less case the transition probabilities are always positive, and for constant volatility σ, shaped like a Laplace density, i.e. $e^{-k|x|}$ for some positive constant k. When the number of time steps is increased then the density converges to a Gaussian shape. The sheet 'implicit 1' can be used to test this.

When the problem has a drift, positive transition probabilities are guaranteed only for the case when dynamic up- and down winding is used for the first derivative. I.e. if (10) is used rather than (9). The sheet 'implicit 2' implements this. If you recalc the sheet, the forward scheme generates transition probabilities for new random vectors r, μ, σ. The produced probabilities are always positive, though not always pretty.

The Crank-Nicolson scheme, $\theta = 0.5$, is also always von Neumann stable and thereby convergent. But even for the zero drift case, the transition probabilities will be non-negative only if the condition (23) is satisfied for the first explicit half time step. The transition probabilities oscillate but converge to a smooth positive Gaussian form as you increase the number of time steps while keeping the maturity constant. The sheet 'cn 1' shows this. Randall and Tavella (2000) and Andreasen (2011) provide theoretical arguments that if the grid is dimensioned according to the ± 5 standard deviations rule, then the number of time steps should be roughly half of the number of spatial steps: $m \approx n/2$.

Moderate drift is generally not a practical problem for the convergence of any of the finite difference methods. But for problems with very strong drift relative to the diffusion the only bullet proof cure is really dynamic up- and down-winding and/or transforms to remove or reduce the drift.

Accuracy

Using (12) its straightforward to verify that the accuracy of each step in (13) is $O(\Delta t^2)$ for $\theta \neq 1/2$ and $O(\Delta t^3)$ for $\theta = 1/2$. As the solution of (13) is repeated $O(\Delta t^{-1})$ times, the accuracy of the total scheme is $O(\Delta t)$ for the explicit and implicit schemes and $O(\Delta t^2)$ for the Crank-Nicolson scheme.

Taylor expansion can be used to show that

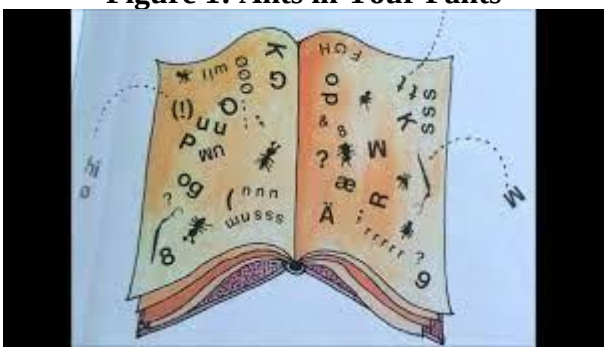
$$\begin{aligned}\delta_x^+ f(x_i) &= f'(x_i) + O(\Delta x) \\ \delta_x^- f(x_i) &= f'(x_i) + O(\Delta x) \\ \delta_x f(x_i) &= f'(x_i) + O(\Delta x^2)\end{aligned} \tag{24}$$

and

$$\delta_{xx} f(x_i) = f''(x_i) + \frac{1}{3} f'''(x_i) \frac{(x_{i+1} - x_i)^2 - (x_i - x_{i-1})^2}{(x_{i+1} - x_{i-1})} + O(\Delta x^2) \tag{25}$$

So in summary, the implicit and explicit schemes have accuracies of $O(\Delta t + \Delta x^2)$, whereas the Crank-Nicolson scheme has an accuracy of $O(\Delta t^2 + \Delta x^2)$. The accuracies drop to respectively $O(\Delta t + \Delta x)$ and $O(\Delta t^2 + \Delta x^2)$ if up- and down-winding is used.

Figure 1: Ants in Your Pants



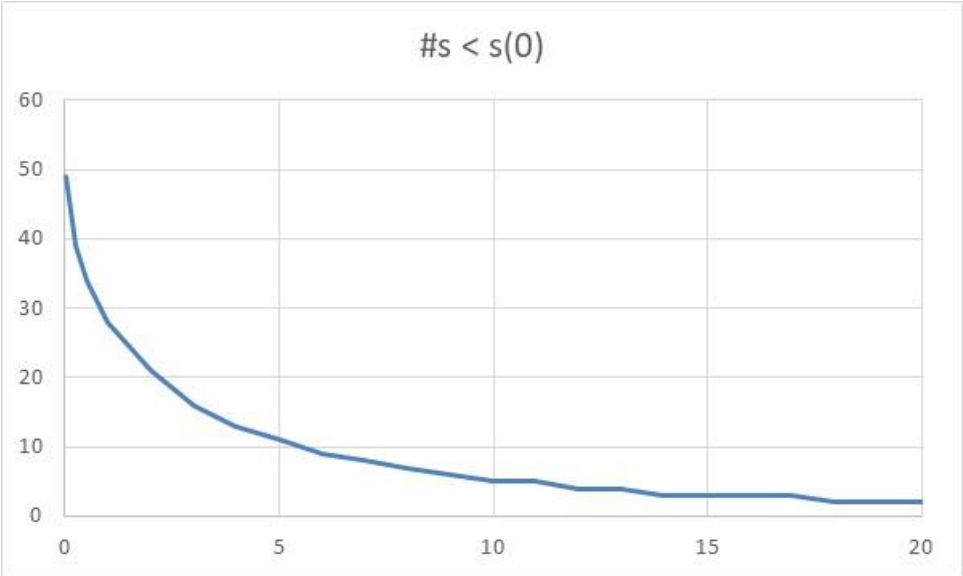
Finite difference code doesn't have to look like this.

Table 2: Option Price as Function of Grid Width

num std	imp vol
1	0.098001088456667
2	0.100039426492236
3	0.100056953428725
4	0.100056956186182
5	0.100056956190353
6	0.100056956190353
7	0.100056956190353
8	0.100056956190353
9	0.100056956190353
10	0.100056956190353

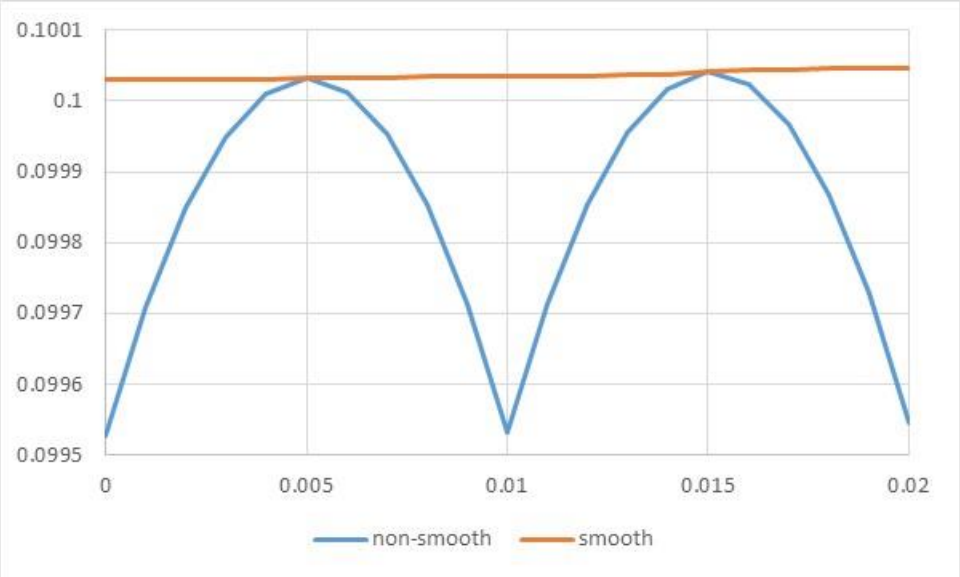
Finite difference solution for different grid width specified in number of standard deviations $q = 1,\ldots,10$ and $\Delta x = 0.02$ held constant. Other parameters are $x(0) = 0, \sigma = 0.1, \mu = r = 0, T = 0.25, k = -0.01, \#\{t_h\} = 10, \theta = 0.5$. See sheet 'grid width'.

Figure 3: Points Below $S(0)$ as Function of T



How many grid points out of total number of $n=100$ grid points is below $S(0)$ when we use constant spacing $\Delta S=(S_{n-1}-S_0)/n$ in a log-normal model with $S(0)=1,\sigma=0.2$, and a grid width of $S_0=e^{-5\sigma\sqrt{T}}, S_{n-1}=e^{5\sigma\sqrt{T}}$. See sheet 'transform'.

Figure 4: Implied Volatility As Function of Strike



Implied volatility of option price as function of strike for non-smoothed and smoothed option payoffs.
Parameters: $T = 0.25, x(0) = 0, r = 0, \mu = 0, \sigma = 0.1, n = 50, \#\{t_h\} = 10, \theta = 0.5$.
See sheet 'smooth'.

References

Andreasen, J (2022a): “By the Rivers of Babylon.” *Wilmott* September.

Andreasen, J (2022b): “Three Strikes and You’re Out.” *Wilmott* November.

Andreasen, J and B Huge (2011a): “Volatility Interpolation.” *Risk* March.

Andreasen, J and B Huge (2011b): “Random Grids.” *Risk* June.

Andreasen, J, B Huge and F Kryger-Baggesen (2022): ”Git Hub Repository.”

Austing, P (2020): “Finite Difference Schemes with Exact Recovery of Vanilla Option Prices.” *Risk* November.

Dupire, B (1994): “Pricing with a Smile.” *Risk* January.