Computational Methods in Finance, Lecture 4, Finite Difference Methods for Diffusion Equations, II.

Jonathan Goodman *
Courant Institute of Mathematical Sciences, NYU

October 9, 1997

1 Introduction

We discuss more technical issues that arise in practice, give alternatives to the forward Euler method, and discuss stability theory for finite difference methods.

2 Non Uniform Meshes

Often it is desirable to use nonuniform grids rather than uniform ones. This means that Δx is not constant. For example, if we are solving the Black Scholes equation with a large volatility or long time to expiration, we might have to use a computational domain with a range of stock prices between 5 and 100, say. A ΔS that is appropriate for $S = 5$ is probably smaller than necessary for $S = 100$. In that case, it makes better sense to keep $\Delta S/S$ constant rather than S constant.

*goodman@cims.nyu.edu, or http://www.math.nyu.edu/faculty/goodman,

If you use a nonuniform mesh, you must find a way to discretize the PDE on the mesh you have. The simplest approach is to use finite difference approximations to derivatives as we have done so far. Unfortunately, it can take considerable effort to make these approximations second order accurate.

There is another approach to discretization on a nonuniform mesh that suffices in many financial applications. This is to use a nonuniform mesh that is uniform with respect to a different variable. For example, in Black-Scholes, we can take

$$S = e^y \quad , \tag{1}$$

and choose a *constant* Δy. Then a uniform grid in the y variable:

$$y_j = j \cdot \Delta y \quad ,$$

would lead to a nonuniform mesh in the S variable:

$$S_j = e^{y_j} = \exp(j\Delta y) \quad . \tag{2}$$

This mesh has the desired property that $\Delta S/S$ is constant. Now, Instead of discretizing the equation using the S variable, we first rewrite the equation in terms of y and discretize this new equation in using a uniform mesh.

It is interesting to work this out for Black-Scholes. We start with the equation

$$\partial_t f + rS\partial_S f + \frac{1}{2}\sigma^2 S^2 f = rm \quad . \tag{3}$$

Chain rule calculations give

$$\frac{\partial f}{\partial S} = \frac{\partial f}{\partial y}\frac{\partial y}{\partial S} = \frac{1}{S}\frac{\partial f}{\partial y} \quad ,$$

and

$$\frac{\partial^2 f}{\partial S^2} = \frac{\partial}{\partial S}\left(\frac{1}{S}\frac{\partial f}{\partial y}\right) = -frac1S^2\frac{\partial f}{\partial y} + \frac{1}{S^2}\frac{\partial^2 f}{\partial y^2} \quad .$$

And so, (3) becomes

$$\partial_t f + \left(r - \frac{\sigma^2}{2}\right)\partial_y f + \frac{1}{2}\sigma^2\partial_y^2 f = rf \quad . \tag{4}$$

We now discretize (4) in the usual way using the forward Euler method. Use the approximations

$$
\begin{aligned}
\partial_y f &\approx \frac{F^n_{j+1} - F^n_{j-1}}{2\Delta y} \ , \\
\partial^2_y f &\approx \frac{F^n_{j+1} - 2F^n_j + F^n_{j-1}}{\Delta y^2} \ , \\
\partial_t f &\approx \frac{F^n_j - F^{n-1}_j}{\Delta t} \ ,
\end{aligned}
$$

substitute into (4), solve for F^{n-1}_j, and you get

$$
\begin{aligned}
F^{n-1}_j &= \left(\frac{\Delta t \sigma^2}{2\Delta y^2} - \left(r - \frac{\sigma^2}{2} \right) \frac{\Delta t}{2\Delta y} \right) F^n_{j-1} \\
&+ \left(1 - \frac{\Delta t \sigma^2}{\Delta y^2} - r\Delta t \right) F^n_j \\
&+ \left(\frac{\Delta t \sigma^2}{2\Delta y^2} + \left(r - \frac{\sigma^2}{2} \right) \frac{\Delta t}{2\Delta y} \right) F^n_{j-1} \ .
\end{aligned}
\tag{5}
$$

This leads to second order accurate approximations to f.

The popular CRR binomial tree method is a special case of (5) for a special choice of Δt To make the middle term on the right vanish, we choose

$$
\Delta t = \frac{1}{\frac{\sigma^2}{\Delta y^2} + r} \ . \tag{6}
$$

Under this restriction, F^{n-1}_j is expressed in terms of the two values F^n_{j+1} and F^n_{j-1}. The coefficient of F^n_{j+1} corresponds to the possibility that "the stock ticks up" in the time step, while the coefficient of F^n_{j-1} corresponds to the possibility that the stock ticks down. The trick (6) leading to a binomial rather than a trinomial tree may not be possible in more complex models. Even when it is possible, it may be undesirable to link the computational mesh to the values of r and σ because (i) we may wish to put an integral number of time steps in time steps in the time interval of the computation while at the same putting an integral number of space steps[1], and (ii) we may wish to compare results for a range of r and σ values on the same mesh.

[1] For example, in valuing barrier options, we often want to put a grid point at the strike price and a grid point at the barrier.

3 Other methods for solving diffusion equations

There are several ways in which other difference approximations to diffusion equations might be preferable. The forward Euler method has a severe time step constraint. Other methods, in particular the backward Euler method and Crank-Nicholson, allow larger Δt and therefore fewer time steps in the computation. Also, there are methods that are more than second order accurate. A higher order of accuracy often means that the computation achieves a given accuracy (1%, say) with fewer grid points. This increase may fail in some financial applications because the initial data or solution are not smooth enough. The smoothness requirements for 4^{th} order are greater than those for second order accuracy. If the "second order" method produces first order results, then the 4^{th} order method probably will also produce first order results.

Many possible discretizations, including forward and backward Euler and Crank-Nicholson, are "method of lines" methods. The "method of lines" is to discretize first in space, leaving a collection of ODE's, and then to use an ODE discretization method to discretize the ODE.

We write an ODE as

$$\dot{x} = f(x) \quad , \tag{7}$$

where $x = (x_1, \ldots, x_n)$. Note that here x is the dependent variable being computed; the notation is different from that of the diffusion equation in this regard. We call the approximate solution $X^n \approx x(n\Delta t)$. The forward Euler approximation to (9) is (check any undergraduate ODE book)

$$X^{n+1} = X^n + \Delta t f(X^n) \quad . \tag{8}$$

Other methods include backward Euler

$$X^{n+1} = X^n + \Delta t f(X^{n+1}) \quad , \tag{9}$$

the trapezoid rule

$$X^{n+1} = X^n + \frac{\Delta t}{2} \Big(f(X^{n+1}) + f(X^n) \Big) \quad , \tag{10}$$

and fancier methods such as Runge-Kutta and linear multistep methods. You can get oriented to this huge field using the recent book of Iserles. Note that the approximations (9) and (10) are "implicit": you have to solve a system of equations to compute X^{n+1} from X^n. This extra complexity is compensated for by gains inefficiency or accuracy.

In order to use a method of lines, we must approximate the PDE with a system of ODE's. This "spatial discretization" is usually straightforward: we just use finite difference approximations to spatial derivatives. Take, for example, the heat equation

$$\partial_t u = \frac{1}{2}\partial_x^2 \ . \tag{11}$$

We choose Δx and write $U_j(t) \approx u(x_j, t)$ for the approximation at grid point x_j. If we use

$$\partial_x^2 u \approx \frac{U_{j+1} - 2U_j + U_{j-1}}{\Delta x^2} \ ,$$

we approximate (11) by

$$\dot{U}_j = \frac{U_{j+1} - 2U_j + U_{j-1}}{\Delta x^2} \ . \tag{12}$$

If we write $U(u)$ for the vector whose components are the $U_j(t)$, then (12) is an ODE of the type (7) with an $f(U)$ that happens to be linear in this case. The forward Euler method (8), when written in coordinates, is

$$U_j^{nm+1} = U_j^n + \frac{\Delta t}{2\Delta x^2}\left(U_{j+1}^n - 2U_j^n + U_{j-1}^n\right) \ ,$$

the same forward Euler we have been using. The backward Euler method (9) becomes

$$U_j^{n+1} - \frac{\Delta t}{2\Delta x^2}\left(U_{j+1}^{n+1} - 2U_j^{n+1} + U_{j-1}^{n+1}\right) = U_j^n \ . \tag{13}$$

This is the backward Euler method for the heat equation. Finally, the trapezoid rule (10) becomes (after some algebra)

$$U_j^{n+1} - \frac{\Delta t}{4\Delta x^2}\left(U_{j+1}^{n+1} - 2U_j^{n+1} + U_{j-1}^{n+1}\right) = U_j^n + \frac{\Delta t}{4\Delta x^2}\left(U_{j+1}^n - 2U_j^n + U_{j-1}^n\right) \ . \tag{14}$$

In the context of diffusion equations, this is called the Crank-Nicholson method.

There are other methods that do not fit this pattern. One such is an application of an idea of Lax and Wendroff to diffusion equations. By differentiating (11) with respect to x and t and manipulation, we get

$$\partial_t^2 u = \frac{1}{4}\partial_x^4 u \ . \tag{15}$$

Thus,

$$u(x,t+\Delta t) \approx u(x,t) + \frac{\Delta t}{2}\partial_x^2 u(x,t) + \frac{\Delta t^2}{8}\partial_x^4 u(x,t) \ . \tag{16}$$

If we take (16) but use a 4^{th} order approximation to $\partial_x^2 u$, we get a 4^{th} order approximation to (11).

The implicit methods (13) and (14) require us to solve a system of equations at each time step. for one factor models, this is easily done using Gaussian elimination for banded matrices. For multifactor models, the task of solving the system of equations requires much more attention but need not increase the running time of the code outrageously.

4 Fourier Stability Analysis

One way a numerical method can fail is by being inaccurate, the exact solution not approximately satisfying the difference equations. A method may also fail by being *unstable*. An unstable method amplifies small errors that inevitably enter the computation until they are larger than the solution being computed. Errors due to instability rarely go unnoticed, unless you don't look at the results.[2]

Stability analysis is generally much harder than accuracy analysis. The only general method applies only to constant coefficient problems with simple boundary conditions. Fortunately, the conclusions of this idealized stability analysis on model problems generally apply to real variable coefficient problems, provided that you satisfy the stability constraint at every point in the computational domain.

The method for this stability analysis is Fourier analysis (often called "von Neumann analysis, after John von Neumann, the Hungarian-American mathematician who first applied it in this context). Before applying this to finite difference methods, we first do it for the heat equation. We say that $u(x)$ is periodic with period 2π if $u(x+2\pi) = u(x)$ for all x. A periodic solution of the heat equation is said (somewhat incorrectly) to satisfy "periodic boundary

[2]This is possible if the computer solved thousands of diffusion equation in an unattended overnight run.

conditions". A periodic function may be represented as a Fourier series

$$u(x) = \sum_{p\infty}^{\infty} \hat{u}_p e^{ipx} \quad .$$

If u depends on t, then all the Fourier coefficients do also:

$$u(x,t) = \sum_{p\infty}^{\infty} \hat{u}_p(t) e^{ipx} \quad . \tag{17}$$

If we substitute the expansion (17) into the heat equation (11) and identify coefficients of e^{ipx}, we get

$$\partial_t \hat{u}_p(t) = \frac{-p^2}{2} \hat{u}_p \quad .$$

This gives the solution formula for (11):

$$u(x,t) = \sum_p e^{-p^2 t/2} \hat{u}_p(0) e^{ipx} \quad , \tag{18}$$

which contains much qualitative information about the solution. The PDE book by Fritz John is a good source for this.

The Fourier analysis of the forward Euler method is slightly more complicated. Suppose that $\Delta x = 2\pi/L$ and that the U_j^n are periodic in the sense that

$$U_{j+L}^n = U_j^n \tag{19}$$

for all j and n. Then corresponding to (17) we have

$$U_j^n = \sum_{p=-L/2}^{L/2} \hat{U}^n e^{ipx_j} \quad . \tag{20}$$

There should be L terms in the sum, corresponding to the L degrees of freedom in U^n. Now substitute (20) into the forward Euler equations and equate coefficients of e^{ipx_j} and you get

$$\hat{U}_p^{n+1} = \left(1 - \frac{\Delta t}{\Delta x^2}(1 - \cos(p\Delta x))\right) \hat{U}_p^n \quad . \tag{21}$$

The "amplification factor" is the coefficient of $\hat{U}_p^n$, namely

$$a_p = 1 - \frac{\Delta t}{\Delta x^2}(1 - \cos(p\Delta x)) \ \ .$$

From (21) we find

$$\hat{U}_p^n = a_p^n \hat{U}_p^0 \ \ . \tag{22}$$

This is the analogue of our formula (18) for the heat equation. The p^{th} Fourier coefficient is stable if $|a_p| \leq 1$ and unstable if $|a_p| > 1$. In the unstable case, (22) shows that $\hat{U}_p^n$ grows exponentially with n, leading to catastrophe if there are more than a few time steps. In the case of forward Euler, $1 - \cos(p\Delta x)$ ranges from 0 to 2, so a_p ranges from 1 to $1 - 2\Delta t/\Delta x^2$. Clearly, we get $|a_p| \leq 1$ for all p if and only if $\Delta t/\Delta x^2 \leq 1$. This is the stability limit we gave before.

For backward Euler, a similar computation leads to

$$\_p = \frac{1}{1 + \frac{\Delta t}{\Delta x^2}(1 - \cos(p\Delta x))} \ \ . \tag{23}$$

Therefore, a_p is between 1 and $1/(1 + 2\Delta t/Deltax^2)$. In particular, $|a_p| \leq 1$ for any p and $\Delta t/\Delta x^2$. This shows that backward Euler is always stable.

For Crank Nicholson, we get

$$a_p = \frac{1 - \frac{\Delta t}{2\Delta x^2}(1 - \cos(p\Delta x))}{1 + \frac{\Delta t}{2\Delta x^2}(1 - \cos(p\Delta x))} \ \ , \tag{24}$$

which also has $|a_p| \leq 1$ for all p. However, there is a difference between (23) and (24) that affects the "highest spatial frequencies", those with $\cos(p\Delta x) \approx -1$. If $\Delta t/\Delta x^2$ is very large, then (23) gives an $|a_p|$ that is very small, while (24) has $a_p \approx -1$. That is, under Crank-Nicholson, the highest Fourier modes do not decay as they should under (18). Rather, they just "ring", switching sigh each time step. This often leads to visible high frequency noise in computations with singular initial data and large time steps.