The Crank-Nicolson Scheme

The fully implicit scheme is unconditionally stable, but it tends to oversuppress short-length-scale fluctuations. It turns out that an even better approach is to take the "average" of the explicit and implicit schemes—a so-called *semi-implicit* scheme, also called the *Crank-Nicolson* scheme:

$$u_j^{n+1} = u_j^n + \tfrac{1}{2}\alpha(u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1} + u_{j+1}^n - 2u_j^n + u_{j-1}^n)\,,$$

or, rearranging,

$$-\tfrac{1}{2}\alpha u_{j+1}^{n+1} + (1+\alpha)u_j^{n+1} - \tfrac{1}{2}\alpha u_{j-1}^{n+1} = \tfrac{1}{2}\alpha u_{j+1}^n + (1-\alpha)u_j^n + \tfrac{1}{2}\alpha u_{j-1}^n\,.$$

This method is widely used because it is unconditionally stable but does not damp the essential features in the solution. Applying the von Neumann stability analysis, we find that

$$\xi = \frac{1 - 2\alpha\sin^2\frac{1}{2}k\Delta x}{1 + 2\alpha\sin^2\frac{1}{2}k\Delta x},$$

from which it is easily seen that $|\xi| < 1$ always, so the method is unconditionally stable. Furthermore, for the scales of greatest interest — comparable to the scale of the grid — we have $k\Delta x \ll 1$ and

$$\xi \;\approx\; 1 - \alpha(k\Delta x)^2 \;\approx\; 1,$$

so damping is minimal. We will also see that the Crank-Nicolson scheme preserves unitarity in the Schrödinger-equation version, making it particularly useful for quantum-mechanical applications.

As with the fully implicit method, the above equation is a tridiagonal matrix system

$$\mathbf{A}\mathbf{x} = \mathbf{r}\,,$$

where (apart from the first and last elements) $x_j = u_j^{n+1}$, $r_j = \frac{1}{2}\alpha u_{j+1}^n - \alpha u_j^n + \frac{1}{2}\alpha u_{j-1}^n$, and

$$\mathbf{A} = \begin{pmatrix} ? & ? & 0 & 0 & \cdots \\ -\frac{1}{2}\alpha & 1+\alpha & -\frac{1}{2}\alpha & 0 & \cdots \\ 0 & -\frac{1}{2}\alpha & 1+\alpha & -\frac{1}{2}\alpha & \cdots \\ \cdot & \cdot & & & \\ \cdot & \cdot & & & \\ \cdot & \cdot & & & \end{pmatrix}.$$

We solve it just as described previously for the fully implicit method. The only differences are the details of the matrix A and the form of $\mathbf{r}$. Thus, during each step, the new u_j array is the solution x_j of the matrix equation

$$\begin{pmatrix} b_0 & c_0 & 0 & 0 & 0 & \cdots \\ a_1 & b_1 & c_1 & 0 & 0 & \cdots \\ 0 & a_2 & b_2 & c_2 & 0 & \cdots \\ \cdot & \cdot & & & & \\ \cdot & \cdot & & & & \\ \cdot & \cdot & & & & \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \end{pmatrix} = \begin{pmatrix} r_0 \\ r_1 \\ r_2 \\ \cdot \\ \cdot \\ \cdot \end{pmatrix},$$

where as usual the top and bottom rows are used to apply the boundary conditions on u and, for $1 \le j \le J-2$, we have $a_j = c_j = -\frac{1}{2}\alpha$, $b_j = 1+\alpha$, and $r_j = \frac{1}{2}\alpha u_{j+1}^n + (1-\alpha)u_j^n + \frac{1}{2}\alpha u_{j-1}^n$.