1 Implicit Methods

The methods we've used so far to solve partial differential equations are called *explicit* methods. These methods have several disadvantages, especially with stability. In addition, explicit methods have low accuracy in time, being order $O(\Delta t)$, and so small time steps are required. This then means many more steps are required.

An alternative approach to solving PDEs numerically exist. These types of methods are called *implicit methods*. We introduce as simple implicit scheme here using the heat equation as a specific example.

The 1-D heat equation is (note, we are not assuming the *steady state here*)

$$u_t = ku_{xx} \tag{1}$$

where k is the heat conductivity. We use the notation

$$u(x_i, t_m) \equiv u_i^m.$$

Discretizing the partials gives

$$\begin{aligned} u_{xx} &\approx \frac{u_{i+1}^{m+1} - 2u_i^{m+1} + u_{i-1}^{m+1}}{h_x^2} \\ u_t &\approx \frac{u_i^{m+1} - u_i^m}{\delta_t} \end{aligned}$$

The term h_x is the grid spacing in space and δ_t the spacing in time. Notice the spatial derivatives all have terms occurring at the same time $m+1$, and at neighboring positions , while the time derivative occurs at the same spatial point, i, but at neighboring time steps.

Adding the R.H.S yields,

$$-\lambda u_{i-1}^{m+1} + (1+2\lambda)u_i^{m+1} - \lambda u_{i+1}^{m+1} = u_i^m \text{ where } \lambda = k\frac{\delta_t}{h_x^2}. \tag{2}$$

In order to apply Eq.(2) two things are required,

- Initial conditions for the case $m = 0$. That is, we need u_i^0 for all i.

- Boundary conditions at $i = 0, i = N - 1$ for all times m.

Implicit schemes like the one outlined in Eq.(2) are unconditionally stable. Because this scheme is order δ_t^2 accurate, one can take fewer time steps. However, the calculations become more difficult when dealing with 2 and 3-D problems.

(1) You will apply Eq.(2) to a 1-D bar with a total four interior spatial points and four time points. The left boundary is set to 100^{o}C and the right boundary is set to 50^{o}C. The temperature is 0 at all other points at $t = 0 (m = 0)$. Using $\lambda = 0.4$, write Eq.(2) with $i = 1$, putting all unknown quantities on the L.H.S. and known quantities on the R.H.S. What interior points are located on the L.H.S.?

(2) Do the same for the interior points $i = 2$ and $i = 3$, again, at time step $t = 0 (m = 0)$.

(3) And finally repeat step (1) at the right boundary, ($i = 4$).

Parts (1) - (3) above develop a simple implicit scheme. The more usual approach in using implicit methods is a scheme called *Crank-Nicolson* method.

Essentially Eq.(2) is replaced with second order in accuracy in both space and time. The central differencing is now

$$
\begin{aligned}
u_{xx} &= \frac{1}{2}\left[\frac{u^m_{i-1} - 2u^m_i + u^m_{i+1}}{h_x^2} + \frac{u^{m+1}_{i-1} - 2u^{m+1}_i + u^{m+1}_{i+1}}{h_x^2}\right] && (3)\\
u_{tt} &= \frac{u^{m+1}_i - u^m_i}{\delta_t} && (4)
\end{aligned}
$$

Substituting these into the heat equation yields,

$$
-\lambda u^{m+1}_{i-1} + 2(1+\lambda)u^{m+1}_i - \lambda^{m+1}_{i+1} = \lambda u^m_{i-1} + 2(1+\lambda)u^m_i - \lambda^m_{i+1} \qquad (5)
$$

Putting these equations in matrix form, we again get a tridiagonal matrix of the form

$$
\begin{pmatrix}
2+2\lambda & -\lambda & 0 & \cdots & 0\\
-\lambda & 2+2\lambda & -\lambda & \dots & 0\\
0 & \vdots & \vdots & \vdots & \\
0 & \vdots & -\lambda & 2+2\lambda & -\lambda\\
0 & \dots & 0 & -\lambda & 2+2\lambda
\end{pmatrix}
\begin{pmatrix}
u^{m+1}_1\\
u^{m+1}_2\\
\vdots\\
\vdots\\
u^{m+1}_{N-1}
\end{pmatrix}
=
$$

$$
\begin{pmatrix}
\lambda u^m_o + (2-2\lambda)u^m_1 + \lambda u^m_2 + \lambda u^{m+1}_o\\
\lambda u^m_1 + (2-2\lambda)u^m_2 + \lambda u^m_3\\
\vdots\\
\vdots\\
\lambda u^m_{N-2} + (2-2\lambda)u^m_{N-1} + \lambda^m_N + \lambda u^{m+1}_N
\end{pmatrix} \qquad (6)
$$

On the right hand side, the terms $u^m_o, u^{m+1}_0, u^m_N, u^{m+1}_N$ are all given by the initial and boundary conditions.

4. Discuss in your room how Eq.(6) works and/or what it means.

5. Download the files `MyCN.m` and `trisolve.m` Teams course page. Make sure you understand how or if the code works. Run the code as is. Then change the initial condition to $u(x,0) = \sin(\pi * x/L)$ where L is the length of the bar. Plot the temperature as a function of time. Does your solution reach the steady state quickly?