3 Explicit versus implicit Finite Difference Schemes

During the last lecture we solved the transient (time-dependent) heat equation in 1D

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2} \tag{1}$$

In *explicit* finite difference schemes, the temperature at time $n+1$ depends explicitly on the temperature at time n. The explicit finite difference discretization of equation 1 is

$$\frac{T_i^{n+1} - T_i^n}{\Delta t} = \kappa \frac{T_{i+1}^n - 2T_i^n + T_{i-1}^n}{\Delta x^2} \tag{2}$$

This can rearranged in the following manner (with all quantities at time $n+1$ on the left-hand-side and quantities at time n on the right-hand-side)

$$T_i^{n+1} = T_i^n + \kappa \Delta t \frac{T_{i+1}^n - 2T_i^n + T_{i-1}^n}{\Delta x^2} \tag{3}$$

Since we know T_{i+1}^n, T_i^n and T_{i-1}^n, we can compute T_i^{n+1}. This is schematically shown on figure 1. The major advantage of explicit finite difference methods is that they are relatively simple and computationally fast. However, the main drawback is that stable solutions are obtained only when

$$0 < \frac{\kappa \Delta t}{\Delta x^2} < 0.5 \tag{4}$$

If this condition is not satisfied, the solution becomes unstable and starts to wildly oscillate.

In *implicit* finite difference schemes, the spatial derivatives $\frac{\partial^2 T}{\partial x^2}$ are evaluated (at least partially) at the new timestep. The simplest implicit discretization of equation 1 is

$$\frac{T_i^{n+1} - T_i^n}{\Delta t} = \kappa \frac{T_{i+1}^{n+1} - 2T_i^{n+1} + T_{i-1}^{n+1}}{\Delta x^2} \tag{5}$$

This can be rearranged so that unknown terms are on the left and known terms are on the right

$$-sT_{i+1}^{n+1} + (1+2s)T_i^{n+1} - sT_{i-1}^{n+1} = T_i^n \tag{6}$$

were $s = \kappa \Delta t / \Delta x^2$. Note that in this case we no longer have an explicit relationship for T_{i-1}^{n+1}, T_i^{n+1} and T_{i+1}^{n+1}. Instead, we have to solve a linear system of equations (now you know why we repeated linear algebra in the first lesson...). The main advantage of implicit finite difference methods is that there are no restrictions on the timestep, which is good news if we want to simulate geological processes at high spatial resolution. Taking large time steps, however, may result in an inaccurate solution. Therefore it is always wise to check the results by decreasing the timestep until the solution doesn't change anymore (this is called converge check).

The implicit method described in equation 6 is second order accurate in space but only first order accurate in time (i.e., $O(\Delta t, \Delta x^2)$). It is also possible to create a scheme which is second order accurate both in time and in space (i.e., $O(\Delta t^2, \Delta x^2)$). One such scheme is the Crank-Nicolson scheme (Fig. 1C), which is unconditionally stable (you can take any timestep).

3.0.1 Solving an implicit finite difference scheme

We solve the transient heat equation 1 on the domain $-\frac{L}{2} \leq x \leq \frac{L}{2}$ with the following boundary conditions

$$T(x = -L/2, t) = T_{left} \tag{7}$$
$$T(x = L/2, t) = T_{right} \tag{8}$$

with the initial condition

$$
\begin{aligned}
T(x < -2.5, x > 2.5, t = 0) &= 300 && (9)\\
T(-2.5 \leq x \leq 2.5, t = 0) &= 1200 && (10)
\end{aligned}
$$

As usual, the first step is to discretize the spatial domain with n_x finite difference points. The implicit finite difference discretization of the temperature equation is

$$-sT_{i+1}^{n+1} + (1 + 2s)T_i^{n+1} - sT_{i-1}^{n+1} = T_i^n \tag{11}$$

where $s = \kappa\Delta t/\Delta x^2$. In addition, the boundary condition on the left boundary gives

$$T_1 = T_{left} \tag{12}$$

and the one on the right

$$T_1 = T_{right} \tag{13}$$

Equations 11, 12 and 13 can be written in matrix form as

$$\mathbf{Ac} = \mathbf{rhs} \tag{14}$$

for a six-node grid, the coefficient matrix $\mathbf{A}$ is

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ \text{-s} & (1+2\text{s}) & \text{-s} & 0 & 0 & 0 \\ 0 & \text{-s} & (1+2\text{s}) & \text{-s} & 0 & 0 \\ 0 & 0 & \text{-s} & (1+2\text{s}) & \text{-s} & 0 \\ 0 & 0 & 0 & \text{-s} & (1+2\text{s}) & \text{-s} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{15}$$

the unknown temperature vector $\mathbf{c}$ is

$$\mathbf{c} = \begin{pmatrix} T_1^{n+1} \\ T_2^{n+1} \\ T_3^{n+1} \\ T_4^{n+1} \\ T_5^{n+1} \\ T_6^{n+1} \end{pmatrix} \tag{16}$$

and the known right-hand-side vector $\mathbf{rhs}$ is

$$\mathbf{rhs} = \begin{pmatrix} T_{left} \\ T_2^n \\ T_3^n \\ T_4^n \\ T_5^n \\ T_{right} \end{pmatrix} \tag{17}$$

p.s.: Try to remember what you remember about linear algebra, and verify that the equations above are correct..

Once the coefficient matrix **A**, and the right-hand-side vector **rhs** have been constructed, MATLAB in-built functions can be used to obtain the solution **c**. First, however, we have to construct the matrixes and vectors. The coefficient matrix **A** can be constructed with a simple loop:.

```
A = sparse(nx,nx);
for i=2:nx-1
        A(i,i-1) = -s;
        A(i,i ) = (1+2*s);
        A(i,i+1) = -s;
end
```

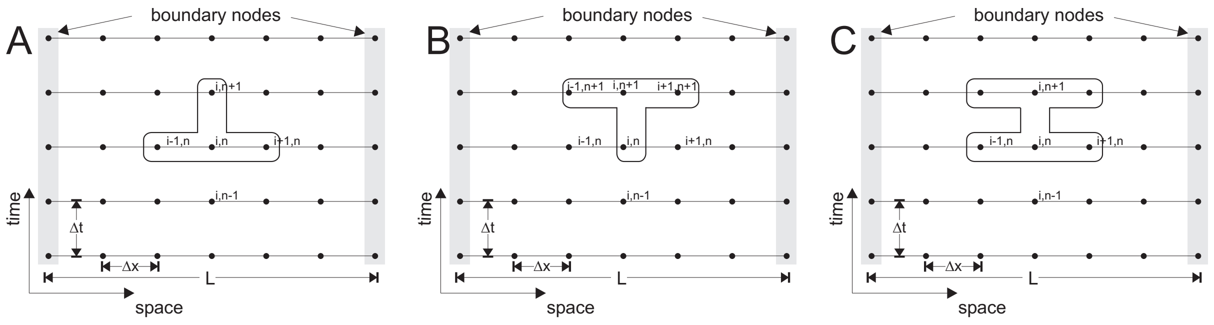

Figure 1: A) Explicit finite difference discretization. B) Implicit finite difference discretization. C) Crank-Nicolson finite difference discretization.

and the boundary conditions are set by:

```
A(1 ,1 ) = 1;
A(nx,nx) = 1;
```

Once the coefficient matrix has been constructed, its structure can be visualized with the command

```
>>spy(A)
```

Note that the matrix is tridiagonal.
The right-hand-side vector **rhs** can be constructed with

```
rhs = zeros(nx,1);
rhs(2:nx-1) = Told(2:nx-1);
rhs(1) = Tleft; rhs(nx) = Tright;
```

The only thing that remains to be done is to solve the system of equations and find **c**. MATLAB does this with

```
c = A\rhs;
```

The vector **c** is now filled with new temperatures T^{n+1}, and we can go to the next timestep. Note that the matrix **A** is constant with time. Therefore we have to form it only once in the program, which speeds up the code significantly. Only the vectors **rhs** and **c** change with time.

4 Exercises

1. Save the script "heat1Dexplicit.m" as "heat1Dimplicit.m". Program the implicit finite difference scheme explained on the previous pages. Compare the results with results from last week's explicit code.

2. A simple (time-dependent) analytical solution for the temperature equation exists for the case that the initial temperature distribution is

$$T(x,t=0) = T_{max}\exp\left[\frac{-x^2}{\sigma^2}\right] \tag{18}$$

where T_{max} is the maximum amplitude of the temperature perturbation at $x = 0$ and σ it's half-width. The solution is than

$$T(x,t) = \frac{T_{max}}{\sqrt{1+4t\kappa/\sigma^2}}\exp\left[\frac{-x^2}{\sigma^2+4t\kappa}\right] \tag{19}$$

Program the analytical solution and compare the analytical solution with the numerical solution with the same initial condition.

3. A steady-state temperature profile is obtained if the time derivative $\frac{\partial T}{\partial t}$ in the temperature equation (eq. 1) is zero. There are two ways to do this. 1). Wait until the temperature doesn't change anymore (it is wise to employ a large timestep dt for this; you can now do this, since the implicit code is stable, independent of timestep. 2. Make a finite difference discretization of $\frac{\partial^2 T}{\partial x^2} = 0$ and solve it. Employ both methods to compute steady-state temperatures for $T_{left} = 100$ and $T_{right} = 1000$.
4. Derive and program the Crank-Nicolson method (fig. 1C).
5. Bonus question 1: write a code for the thermal equation with variable thermal conductivity k: $\rho c_p \frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(k \frac{\partial T}{\partial x}\right)$ Assume that the grid spacing Δx is constant.
6. Bonus question 2: Write a code for the case with variable grid spacing Δx and variable thermal conductivity k.