Numerical methods for solving the heat, Laplace and wave equations (finite difference methods):

Idea: Use finite difference quotients to approximate the derivatives in PDEs.

Example: Remember that the definition of the derivative of a function $f$ is:

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

If $\Delta x$ is sufficiently small, we can approximate the derivative as the finite difference quotient:

$$f'(x) \approx \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

If the function is differentiable many times, you can find the derivative approximations using Taylor series expansion.
\[ f(x + \Delta x) = f(x) + \Delta x f'(x) + \frac{\Delta x^2}{2!} f''(x) + \frac{\Delta x^3}{3!} f^{(3)}(x) + \ldots \]

So, you can find the first derivative approximation by dividing the Taylor expansion above by \( \Delta x \) and re-arranging terms:

\[
f'(x) = \frac{f(x + \Delta x) - f(x)}{\Delta x} - \frac{\Delta x f''(x)}{2!} + \frac{\Delta x^2 f^{(3)}(x)}{3!} + \ldots
\]

\[
\text{truncation errors}
\]

\[ O(\Delta x) : \text{how big the error is.} \]

So, the finite difference approximation:

\[
f'(x) = \frac{f(x + \Delta x) - f(x)}{\Delta x} \quad \text{a forward difference approximation}
\]

has first order accuracy, i.e. the error is a linear function of \( \Delta x \).
How can you have a better accuracy approximation for $f'(x)$?

You have to use more points:

$$f(x + \Delta x) = f(x) + \Delta x f'(x) + \frac{\Delta x^2}{2} f''(x) + \frac{\Delta x^3}{3!} f'''(x) + \ldots$$

$$f(x - \Delta x) = f(x) - \Delta x f'(x) + \frac{\Delta x^2}{2} f''(x) - \frac{\Delta x^3}{3!} f'''(x) + \ldots$$

Subtract the second expansion from the first:

$$f(x + \Delta x) - f(x - \Delta x) = 2\Delta x f'(x) + \frac{2\Delta x^3}{3!} f'''(x) + \ldots$$

$$f'(x) = \frac{f(x + \Delta x) - f(x - \Delta x)}{2\Delta x} - \frac{2\Delta x^2}{3!} f'''(x) + \ldots$$

\[
0(\Delta x^2)
\]

A second order accuracy finite difference approximation.

The error is a quadratic function of $\Delta x$. 

*Delta x*
Second derivative approximation:

To find a finite difference approximation for \( f''(x) \), you need to use two Taylor series expansions:

\[
\begin{align*}
  f(x + \Delta x) &= f(x) + \Delta x f'(x) + \frac{\Delta x^2}{2!} f''(x) + \frac{\Delta x^3}{3!} f'''(x) + \frac{\Delta x^4}{4!} f^{(4)}(x) + \ldots \quad (3) \\
  f(x - \Delta x) &= f(x) - \Delta x f'(x) + \frac{\Delta x^2}{2!} f''(x) - \frac{\Delta x^3}{3!} f'''(x) + \frac{\Delta x^4}{4!} f^{(4)}(x) + \ldots \quad (4)
\end{align*}
\]

Add the two expansions above and divide by \( \Delta x^2 \):

\[
\frac{f(x + \Delta x) - 2f(x) + f(x - \Delta x)}{\Delta x^2} = f''(x) + \frac{2\Delta x^2}{4!} f'''(x) + \ldots \quad (4)
\]

So, the second derivative approximation:

\[
f''(x) = \frac{f(x + \Delta x) - 2f(x) + f(x - \Delta x)}{\Delta x^2} \quad (\text{Central difference approximation})
\]
has a second order accuracy, i.e.

the error is a quadratic function of \( \Delta x \).

Solving the 1D heat equation using finite difference

\[
\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}, \quad 0 < x < 1, \quad t > 0
\]

B.C.'s: \( u(0, t) = 0, \quad u(1, t) = 0 \)

I.C.'s: \( u(x, 0) = f(x) \)

This is an initial-boundary value problem.

The highest order of time derivative is 1

\rightarrow So you need 1 initial condition.

The highest order of spatial derivative is 2,

So you need 2 boundary conditions.

\( u(0, t) = 0 \)

\( u(1, t) = 0 \)

\( u(x, 0) = f(x) \)
First you have to define a discretization is space and time:

- Divide the spatial interval \([0, 1]\) into \(N+1\) equally spaced grid points.

- Divide the time interval of interest \([0, T]\) into \(M+1\) equal time intervals: \(t_k = k \Delta t\).

The value of \(u\) at grid point \(n\), at time step \(k\) is: \(u(x_n, t_k) = u^n_k\).
Now, replace the derivatives in the 1D heat equation by finite difference approximations:

**Time derivative:** forward difference scheme

\[
\frac{\partial u(x,t)}{\partial t} = \frac{u(x,t+\Delta t) - u(x,t)}{\Delta t} + O(\Delta t)
\]

**Space derivative:** central difference scheme

\[
\frac{\partial^2 u(x,t)}{\partial x^2} = \frac{u(x+\Delta x,t) - 2u(x,t) + u(x-\Delta x,t)}{\Delta x^2} + O(\Delta x^2)
\]

Substitute into \( \frac{du}{dt} = \alpha^2 \frac{\partial^2 u}{\partial x^2} \):

\[
\frac{u(x,t+\Delta t) - u(x,t)}{\Delta t} = \alpha^2 \left[ \frac{u(x+\Delta x,t) - 2u(x,t) + u(x-\Delta x,t)}{\Delta x^2} \right].
\]

Now, re-arrange terms to find \( u(x,t+\Delta t) \):

\[
u(x,t+\Delta t) = u(x,t) + \alpha^2 \Delta t \left[ \frac{u(x+\Delta x,t) - 2u(x,t) + u(x-\Delta x,t)}{\Delta x^2} \right].
\]
Or, in the index notation:

\[ u^{n+1} = u^n + \alpha \frac{\Delta t}{\Delta x^2} (u^n - 2u^{n+1} + u^{n-1}) \] \hspace{1cm} (1)

Note that in this numerical scheme to find \( u^{n+1} \) we use information from \( u^n \) and \( u^{n-1} \).

\[ u^{n-1} : \hspace{3cm} u^n : \hspace{3cm} u^{n+1} \]

\[ t_k + \Delta t = t_{k+1} \]

Boundary conditions:

**B.C.** : \( u(0,t) = 0 \), \( u(1,t) = 0 \)

So: \( u^0 = 0 \), \( u^n = 0 \) \hspace{1cm} (2)

These are easy B.C.'s to implement, as at each time step \( k \), we only need to find \( u^k \) to \( u^{k-1} \).
The numerical scheme in (1) and the B.C.'s in (2) can be written in a matrix format as:

\[
\begin{bmatrix}
U_0 \\
U_1 \\
\vdots \\
U_N \\
\end{bmatrix}
= 
\begin{bmatrix}
1 & -1 & 0 & \cdots & 0 \\
\frac{2 \Delta t}{\Delta x^2} & 1 - 2 \frac{\Delta t}{\Delta x^2} & \frac{\Delta t}{\Delta x} & \cdots & 0 \\
0 & \frac{2 \Delta t}{\Delta x^2} & 1 - 2 \frac{\Delta t}{\Delta x^2} & \frac{\Delta t}{\Delta x} & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & 0 & \frac{2 \Delta t}{\Delta x^2} & 1 - 2 \frac{\Delta t}{\Delta x^2} & \frac{\Delta t}{\Delta x} \\
\end{bmatrix}
\begin{bmatrix}
U_0 \\
U_1 \\
\vdots \\
U_N \\
\end{bmatrix}
\]

This is a constant-coefficient matrix \((3)\).

Implementing derivative B.C.'s:

Implementing derivative B.C.'s is more challenging. Assume instead of B.C.'s in (2) you had:

\[U(0, t) = 0, \quad \frac{\partial u}{\partial x}(1, t) = 0\]

Use a central difference approximation to implement \(\frac{\partial u}{\partial x}(1, t) = 0\) at \(x_N = N \Delta x = 1\).
\[ u(x_N + \Delta x, t) - u(x_N - \Delta x, t) \]
\[ \frac{2 \Delta x}{2 \Delta x} = 0 \]

or:
\[ \frac{k}{2 \Delta x} - \frac{k}{2 \Delta x} = 0 \]

This gives a higher accuracy compared to forward difference scheme.

But, we do not have a grid point \( N+1 \) at location \( x_N + \Delta x \). Let's introduce a new grid point \( u_{N+1} \) which has values equal to \( u_{N-1} \) at each time step. We use the same heat equation at \( x_N \):

\[ u^{(N+1)}_N = u^k_N + \Delta t \frac{\Delta x^2}{2} (u^k_{N+1} - 2u^k_N + u^k_{N-1}) \]

(5)
Now, the numerical scheme (1), (5) and the B.C.  
(4) can be summarized in the matrix format as:

$$
\begin{bmatrix}
U^0 \\
U^1 \\
U^n \\
U_{n+1} \\
\end{bmatrix}^{k+1} = 
\begin{bmatrix}
1 & 1 & 0 & 0 & 0 \\
\frac{\partial^2 \phi}{\partial x^2} & 1 - 2\frac{\partial \phi}{\partial x} & \frac{\partial \phi}{\partial x} & & \\
\frac{\partial^2 \phi}{\partial x^2} & 1 - 2\frac{\partial \phi}{\partial x} & \frac{\partial \phi}{\partial x} & & \\
\frac{\partial^2 \phi}{\partial x^2} & 1 - 2\frac{\partial \phi}{\partial x} & \frac{\partial \phi}{\partial x} & & \\
\frac{\partial^2 \phi}{\partial x^2} & 1 - 2\frac{\partial \phi}{\partial x} & \frac{\partial \phi}{\partial x} & & \\
\end{bmatrix} 
\begin{bmatrix}
U^0 \\
U^1 \\
U^n \\
U_{n+1} \\
\end{bmatrix}^k
$$

a constant-coefficient matrix.

See the attached Matlab codes that programs this problem with different B.C.'s.