Consider the one-dimesional heat equation, $u_t = u_{xx}$, $0 < x < 1, t > 0$, subject to homogeneous Dirichlet boundary conditions:

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

and the initial condition:

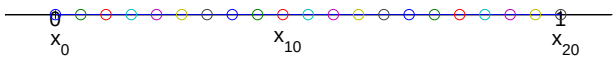
$$u(x,0) = f(x) = \begin{cases} 2x & 0 \le x \le \frac{1}{2} \\ 2-2x & \frac{1}{2} \le x \le 1 \end{cases}$$

(This example is taken from Morton and Mayers' book, pg 11).

Using separation of variables, the exact solution is

$$u(x,t) = \sum_{n=1}^{\infty} c_n e^{-(n\pi)^2 t} \sin(n\pi x), \qquad c_n = 2\int_0^1 f(x)\sin(n\pi x) = \frac{8}{n^2\pi^2}\sin\left(\frac{n\pi}{2}\right), \quad n = 1, 2, \ldots \infty$$

Alternatively, we can *approximate* the solution to a value of $u(x,t)$ using a finite difference scheme. Suppose we divide the interval $[0,1]$ on the x-axis into $N = 20$ intervals and set $x_0 = 0$, $x_1 = h$, $\ldots, x_i = ih$, $\ldots, x_{20} = 1$ with $h = \frac{1}{20}$.



If we set $t_0 = 0$ and let $U_j^m \approx u(x_j, t_m)$, then the explicit finite difference scheme based on centered differences in space and a forward difference in time (see lecture notes) yields 19 equations for approximations to $u(x,t)$ at the interior space nodes, at each time level. We have:

$$U_j^{m+1} = U_j^m + \nu\left(U_{j+1}^m - 2U_j^m + U_{j-1}^m\right), \quad j = 1:19, \quad m = 1, 2, \ldots$$

where $\nu = \frac{k}{h^2}$. The boundary conditions give values for the end points at each time level:

$$U_0^m = U_{20}^m = 0, \quad m = 1, 2, \ldots$$

and the initial conditions give the values for all nodes at the first time level:

$$U_j^0 = f(x_j), \quad j = 0 : N.$$

If we gather the interior values $U_1^m \ldots U_{19}^m$ into a vector $\underline{u}^m$, and note that $U_0^m = f(x_0) = f(0) = 0$ and $U_{20}^m = f(x_{20}) = f(1) = 0$ then, given the initial data $\underline{u}^0$, we can obtain an approximation at the mth time level $t = t_m = mk$, via,

$$\underline{u}^m = T\underline{u}^{m-1}, \quad m = 1, 2, 3, \ldots$$

where T is a 19×19 tri-diagonal matrix (exercise),

$$T = \begin{bmatrix} 1-2\nu & \nu & 0 & 0 & \ldots & 0 \\ \nu & 1-2\nu & \nu & 0 & \ldots & 0 \\ 0 & \nu & 1-2\nu & \nu & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \nu & 1-2\nu & \nu \\ 0 & 0 & 0 & 0 & \nu & 1-2\nu \end{bmatrix}$$

So, at time $t = t_1$, we compute the solution $\underline{u}^1 = T\underline{u}^0$. At $t = t_2$, $\underline{u}^2 = T\underline{u}^1$, etc. Performing matrix-vector products with a large matrix is tedious and best done on a computer. The MATLAB code `heat_eq_explicit_fd`, which you can download from the course webpage, will do this for you. To run the program you must specify three arguments: `N` the number of intervals in the x axis, `k,` the time-step to use, and `t_steps`, the total number of time steps to take. Download the code and try and reproduce the results below.

Example A. Suppose we choose $k = 0.0012$. Then with $N = 20$ (as above), $h = \frac{1}{20}$ and we obtain $\nu = 0.48$. In the table below we list the exact solution to the heat equation with the initial and boundary conditions given above, at the grid points $x_0, \ldots x_{20}$ and the numerical approximations obtained for $u(x,t)$ at those x-points at time levels $t_1 = 0.0012, t_{25} = 0.06$ (after 25 time steps) and $t_{50} = 0.60$ (after 60 time steps).

j	$u(x_j, t_1)$	U_j^1	$u(x_j, t_{25})$	U_j^{25}	$u(x_j, t_{50})$	U_j^{50}
0	0	0	0	0	0	0
1	0.1000	0.1000	0.0915	0.0918	0.0699	0.0699
2	0.2000	0.2000	0.1812	0.1817	0.1382	0.1382
3	0.3000	0.3000	0.2675	0.2684	0.2031	0.2031
4	0.4000	0.4000	0.3484	0.3489	0.2631	0.2631
5	0.5000	0.5000	0.4218	0.4229	0.3167	0.3167
6	0.6000	0.6000	0.4857	0.4860	0.3626	0.3625
7	0.6999	0.7000	0.5381	0.5389	0.3995	0.3994
8	0.7985	0.8000	0.5770	0.5768	0.4267	0.4265
9	0.8843	0.9000	0.6010	0.6016	0.4432	0.4430
10	0.9218	0.9040	0.6091	0.6087	0.4488	0.4486
11	0.8843	0.9000	0.6010	0.6016	0.4432	0.4430
12	0.7985	0.8000	0.5770	0.5768	0.4267	0.4265
13	0.6999	0.7000	0.5381	0.5389	0.3995	0.3994
14	0.6000	0.6000	0.4857	0.4860	0.3626	0.3625
15	0.5000	0.5000	0.4218	0.4229	0.3167	0.3167
16	0.4000	0.4000	0.3484	0.3489	0.2631	0.2631
17	0.3000	0.3000	0.2675	0.2684	0.2031	0.2031
18	0.2000	0.2000	0.1812	0.1817	0.1382	0.1382
19	0.1000	0.1000	0.0915	0.0918	0.0699	0.0699
20	0	0	0	0	0	0

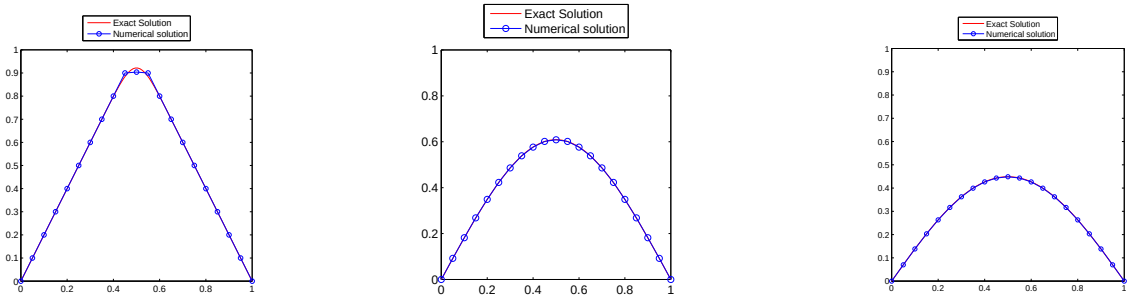


Figure 1: Exact solution and numerical approximations to the solution at time steps: 1, 25, and 50 (left to right)

Note that we do not actually compute the boundary values at each time level, they are given as part of the problem definition. Comparing the three pairs of columns, we see that at each of the time levels considered, the numerical approximation scheme gives a good approximation to the exact solution at each of the grid points.

This is clearly seen in the graphs above. To obtain these results for yourself, run the code. Eg.

```
[u_approx,u_exactx]=heat_eq_explicit_fd(20,0.0012,50);
```

`u_approx` contains the finite difference approximation and `u_exactx` contains the exact solution at the chosen x points. Subtracting the two results gives the error at each point.

Although I don't expect you to perform large matrix vector products by hand to obtain the whole solution at a particular time level, you should practise performing the finite difference calculations by hand for a few points. (You may have to do this in an exam). For instance, in the above example with $\nu = 0.48$ we have:

$$\begin{aligned} U_1^1 &= U_1^0 + 0.0048\left(U_2^0 - 2U_1^0 + U_0^0\right) = f(x_1) + 0.0048\left(f(x_2) - 2f(x_1) + f(x_0)\right) \\ &= f\left(\frac{1}{20}\right) + 0.0048\left(f\left(\frac{2}{20}\right) - 2f\left(\frac{1}{20}\right) + f\left(0\right)\right) = \frac{2}{20} + 0.048\left(\frac{4}{20} - \frac{4}{20} + 0\right) = 0.1 \end{aligned}$$

Example B. Now consider what happens if we make a very small change to our time step k in the approximation scheme. Suppose we choose $k = 0.0013$. Then with the same number of x-points, i.e. with $h = \frac{1}{20}$ we obtain $\nu = 0.52$. In the table below we list the exact solutions at the grid points $x_0, \ldots x_{20}$, the numerical approximations obtained at those x-points at time levels t_1, t_{25} and t_{50} and the absolute values of the errors.

j	$u(x_j,t_1)$	U_j^1	—**error**—	$u(x_j,t_{25})$	U_j^{25}	—**error**—	$u(x_j,t_{50})$	U_j^{50}	—**error**—
0	0	0	0	0	0	0	0	0	0
1	0.1000	0.1000	0.0000	0.0897	0.0930	0.0033	0.0666	0.0467	0.0199
2	0.2000	0.2000	0.0000	0.1777	0.1721	0.0056	0.1316	0.1710	0.0393
3	0.3000	0.3000	0.0000	0.2621	0.2721	0.0100	0.1935	0.1353	0.0581
4	0.4000	0.4000	0.0000	0.3409	0.3287	0.0122	0.2506	0.3259	0.0753
5	0.5000	0.5000	0.0000	0.4123	0.4294	0.0171	0.3016	0.2102	0.0914
6	0.6000	0.6000	0.0000	0.4743	0.4547	0.0195	0.3452	0.4498	0.1046
7	0.6999	0.7000	0.0001	0.5248	0.5482	0.0234	0.3803	0.2640	0.1163
8	0.7981	0.8000	0.0019	0.5623	0.5367	0.0256	0.4060	0.5299	0.1238
9	0.8824	0.9000	0.0176	0.5854	0.6126	0.0272	0.4218	0.2921	0.1296
10	0.9186	0.8960	0.0226	0.5932	0.5652	0.0280	0.4270	0.5576	0.1306
11	0.8824	0.9000	0.0176	0.5854	0.6126	0.0272	0.4218	0.2921	0.1296
12	0.7981	0.8000	0.0019	0.5623	0.5367	0.0256	0.4060	0.5299	0.1238
13	0.6999	0.7000	0.0001	0.5248	0.5482	0.0234	0.3803	0.2640	0.1168
14	0.6000	0.6000	0.0000	0.4743	0.4547	0.0195	0.3452	0.4498	0.1046
15	0.5000	0.5000	0.0000	0.4123	0.4294	0.0171	0.3016	0.2102	0.0914
16	0.4000	0.4000	0.0000	0.3409	0.3287	0.0122	0.2506	0.3259	0.0753
17	0.3000	0.3000	0.0000	0.2621	0.2721	0.0100	0.1935	0.1353	0.0581
18	0.2000	0.2000	0.0000	0.1777	0.1721	0.0056	0.1316	0.1710	0.0393
19	0.1000	0.1000	0.0000	0.0897	0.0930	0.0033	0.0666	0.0467	0.0199
20	0.0000	0	0.0000	0.0000	0	0.0000	0.0000	0	0.0000

After one time step, there is a small error at nodes close to $x = \frac{1}{2}$. As we perform more steps in time, these small errors are amplified and spread to all grid points. After 50 time steps, the numerical approximation is unusable. See the graphs below. To obtain these results for yourself, run the code again changing k to 0.0013. Eg. to generate the approximation at t_{50} use:

```
[u_approx,u_exactx]=heat_eq_explicit_fd(20,0.0013,50);
```

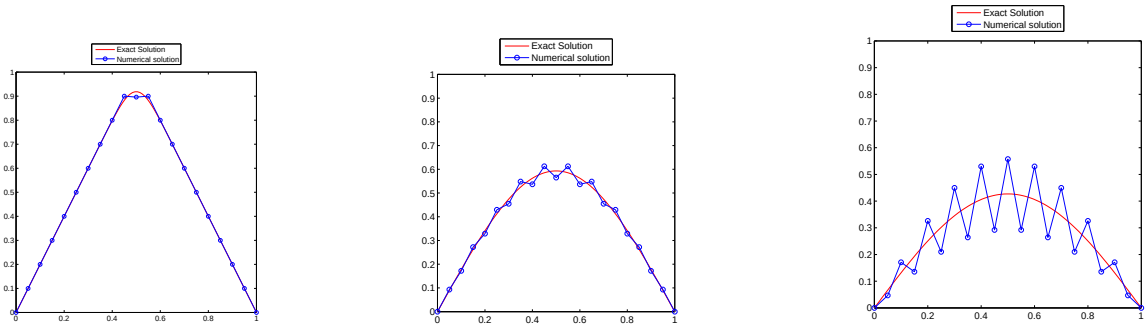


Figure 2: Exact solution and numerical approximations to the solution at time steps: 1, 25, and 50 (left to right)

You should compare the results of these two experiments with the theory discussed in lectures. In the first example, the stability restriction on the finite difference scheme is satisfied. In example B, it is not. The time-step chosen is too large.