Coding Project 3

Irene Natale - in2582

1 Benchmark

the aim of this coding project is to take the Matlab code that we already had and adapt it in a way that it is possible to solve a time-dependent parabolic differential equation. In practice, we will do the same computation as last time for every time step, and we will use Backward Euler methods to "unify" the pieces in the time direction development of the system. We first do some benchmark computations and we use our algorithm for the resolution of an equation for which we know the solution. Knowing the analytic solution $p(x, y, t) = \cos(x - y + t)$ allows us to derive the complete governing system just substituting it into the system equations. The governing system is:

$$\frac{\partial p}{\partial t} - \nabla \cdot (K\nabla p) = q \quad (x, y) \in (0, 1)x(0, 1)x(0, T]$$

$$p = g_D(x, t) \quad x \in \Gamma_D, t \in (0, T]$$

$$u \cdot n = g_N(x, t) \quad x \in \Gamma_N, t \in (0, T]$$

where $u = -K\nabla p$ The initial conditions, boundary conditions and source functions are the following

$$q = -\sin(x - y + t) + (x + y)\cos(x - y + t)$$

$$g_D = \cos(x + t) \quad (x, y) \in (\cdot, 0)$$

$$g_D = \cos(1 - y + t) \quad (x, y) \in (1, y)$$

$$g_N = -y\sin(-y + t) \quad (x, y) \in (0, y)$$

$$g_N = -x\sin(x - 1 + t) \quad (x, y) \in (x, 1)$$

$$p(x, y, 0) = \cos(x - y)$$

In Figure 1, 2 and 3 it is possible to observe the numerical solution compared with the analytic solution of the problem presented in part (a) of the coding project. It is possible to observe that the two solutions have the same behaviour, so we can conclude that the finite element method code works.
Figure 4 shows the 3d solution plot for a mesh of 64*64 elements. Figure 5 shows a 2d coloured plot for a 10*10 elements plot.
In table 1 the results for the space convergence rate are shown. It is possible to observe that the value of the rate is growing at every step, and it is nearer to 2 (the expected space convergence rate for the finite element method with quadratic shape functions).
In table 2 instead the results for the time convergence are shown. It is possible to notice that the time convergence rate goes to the value of 1, which is what it was expected from applying Backward Euler.

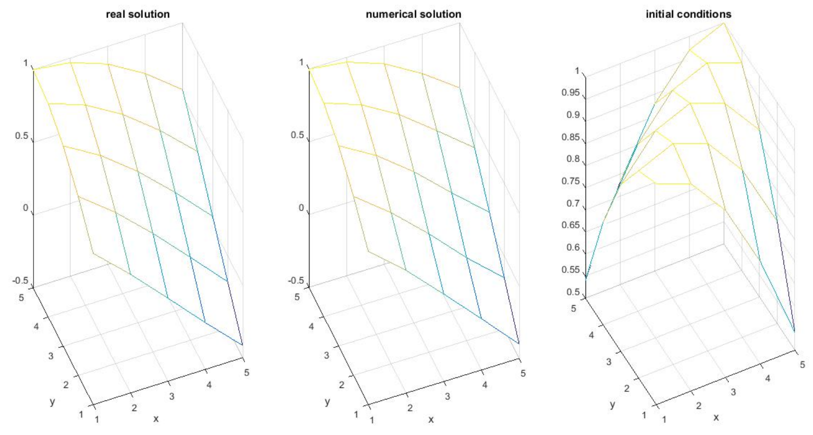

Figure 1: 4*4 elements mesh solution

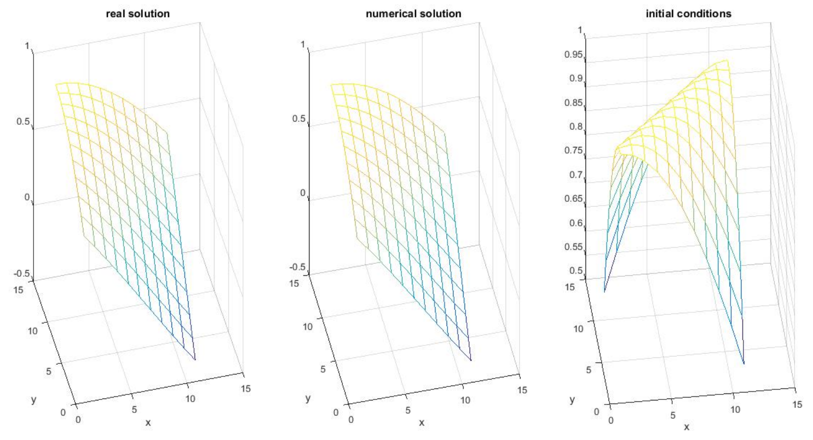

Figure 2: 10*10 elements mesh solution

Table 1: Benchmark results for space

cycle	dt	mesh size	#cells	#dofs	L2-error	space conv rate
1	0.0025	0.25	4*4	5*5 - 5*4+4	0.0338	
2	0.0025	0.125	8*8	9*9-9*4+4	0.0192	0.815917
3	0.0025	0.1	10*10	11*11-11*4+4	0.0145	1.2582
4	0.0025	0.8333	12*12	13*13-13*4+4	0.0112	1.41637

Table 2: Benchmark results for time

cycle	dt	mesh size	#cells	#dofs	L2-error	temp conv rate
1	0.25	0.05	20*20	20*20-20*4+4	0.3537	
2	0.1	0.05	20*20	20*20-20*4+4	0.1849	0.708
3	0.0666666	0.05	20*20	20*20-20*4+4	0.1263	0.9398
4	0.05	0.05	20*20	20*20-20*4+4	0.0965	0.9355

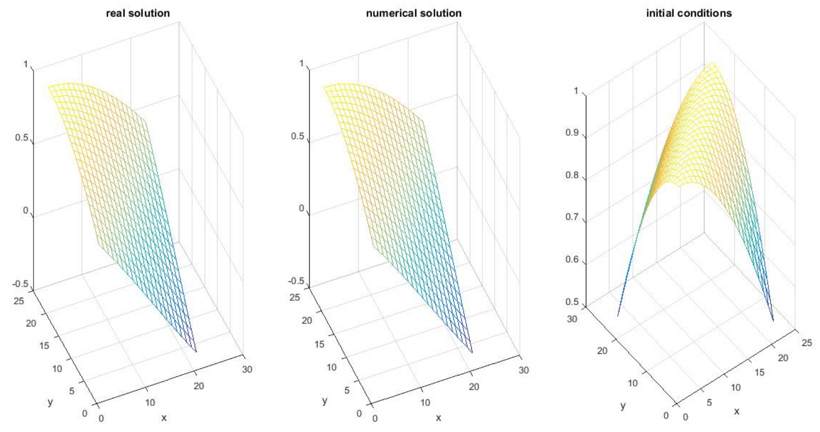

Figure 3: 20*20 elements mesh solution

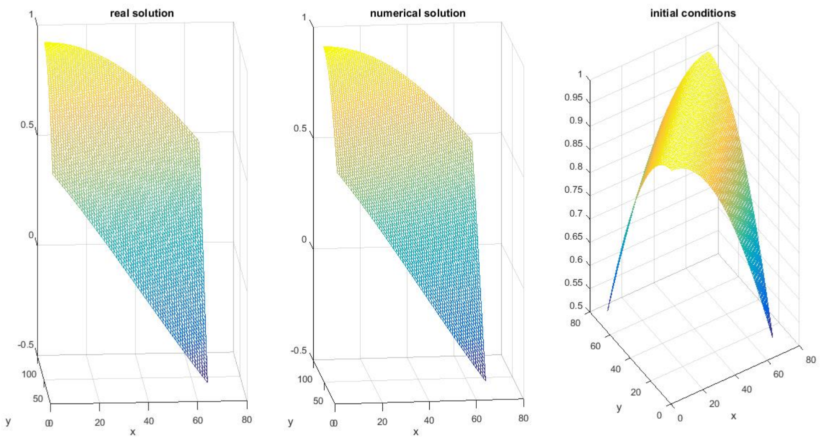

Figure 4: 64*64 elements mesh solution

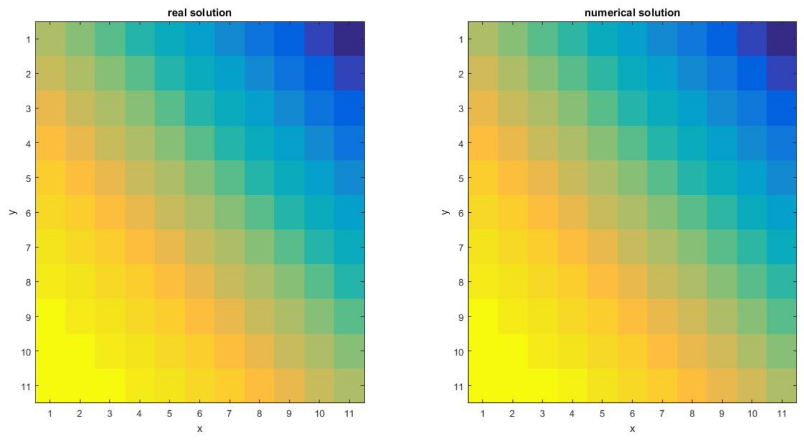

Figure 5: 10*10 elements mesh 2d-solution

2 Application

For the second given problem everything becomes more complicated. The main complication is that we can not check if the final numerical solution is correct because we don't have the analytic solution. The second problem is that the initial condition is described by a delta di Dirac function, and this is hard to handle for the presented code because no integration nodes are on (0.5, 0.5). This can be solved in two ways.
The first solution can be modify the source function in a way that with a fine spacial grid, the delta di Dirac values are considered in the integration. The plot from this solution is shown in Figure 6. This does not lead to an accurate approximation of the problem, because the initial conditions do not reflex the actual behaviour of the system.
A second way to solve this problem is simply assign to the central node of the grid the value of the delta of Dirac function. In this way the initial conditions will not come from the actual source function but they will be directly assigned. The solution plot for this case is shown in Figure 7 and 8 (with meshes of 4*4 and 10*10 elements respectively, at time t=1). Figure 9 instead shows the 2d coloured plot of the solution for a 10*10 mesh at time t=1. Finally, Figure 10 shows the behaviour at $x_2 = 0.5$ along time. It is possible to notice that the curve grows higher when time passes by, and this is due to the heat source that remains at the constant value on 2 after t=0.01.

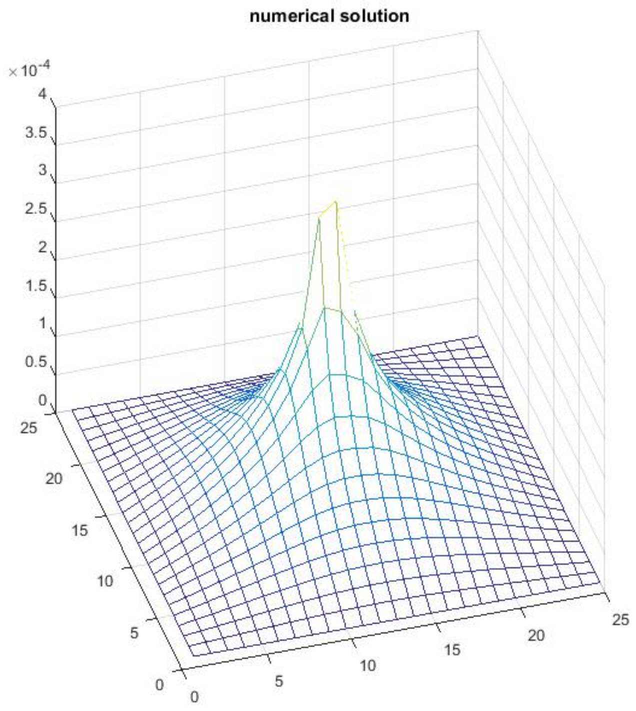

Figure 6: 25*25 elements mesh solution at t=0.1

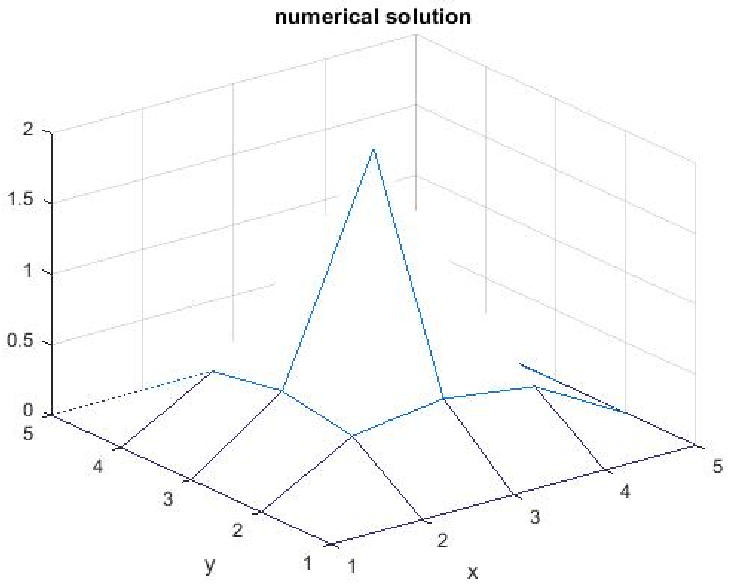

Figure 7: 4*4 elements mesh solution at t=1

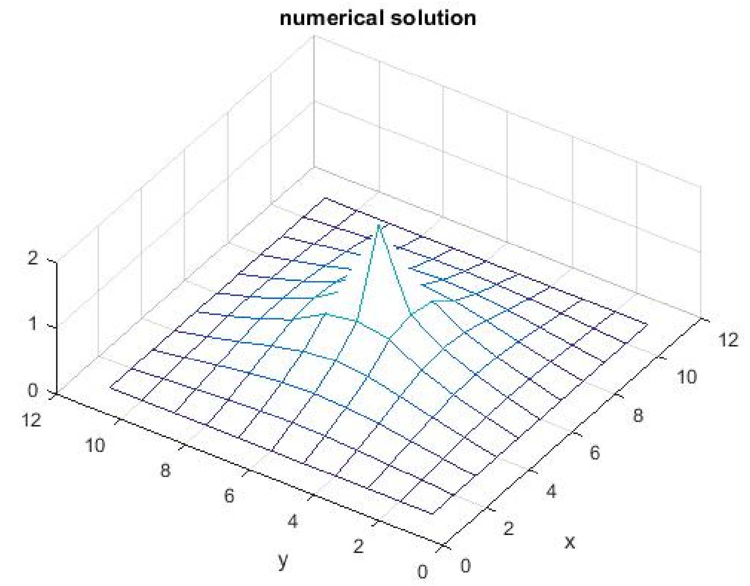

Figure 8: 10*10 elements mesh solution at t=1

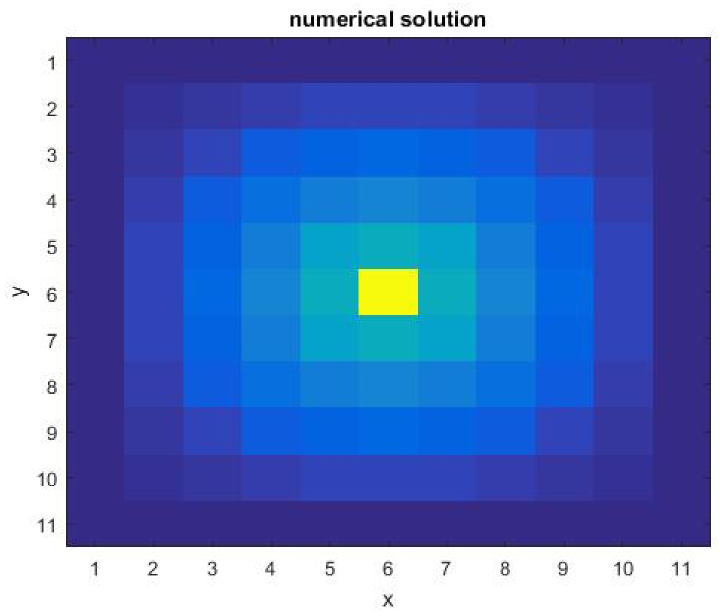

Figure 9: 10*10 elements mesh solution at t=1

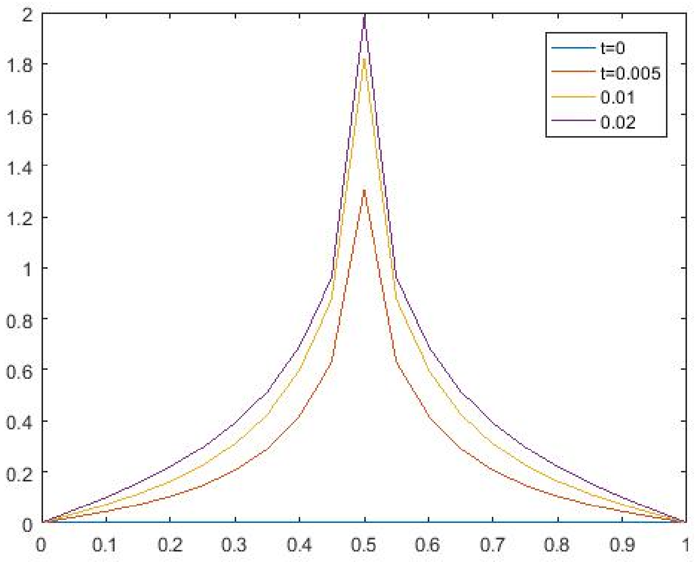

Figure 10: solution at $x_2 = 0.5$ for different time moments