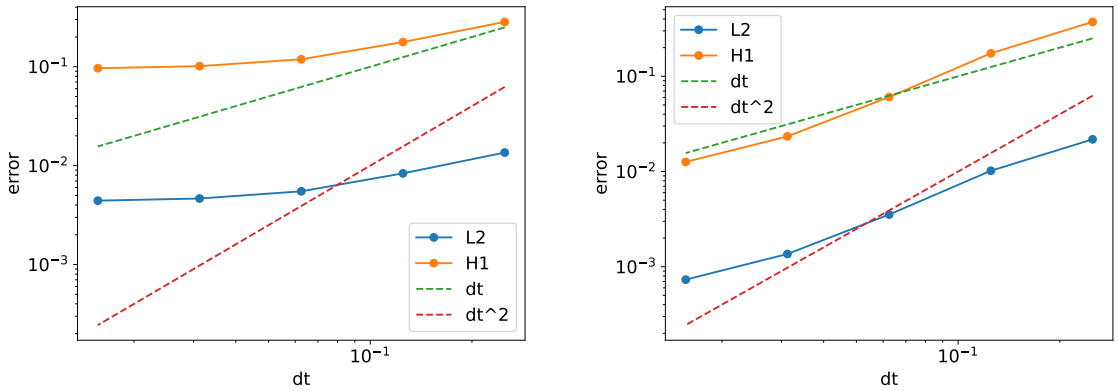


(a) (b)

Figure 1: Error in the L^2 and H^1 norms against Δt, using linear (a) and quadratic (b) polynomials on mesh `mesh/mesh-cube-10.msh` using the implicit Euler method ($\theta = 1$).

1.5. Repeat Points 3 and 4 using the mesh `mesh/mesh-cube-20.msh` and using the Crank-Nicolson method (i.e. setting $\theta = \frac{1}{2}$).

Solution. The resulting errors are reported in Figure 2. With linear polynomials, we observe a convergence order smaller than the expected one ($q = 2$) for the Crank-Nicolson method. Conversely, by increasing the polynomial order, we recover the expected quadratic convergence rate.

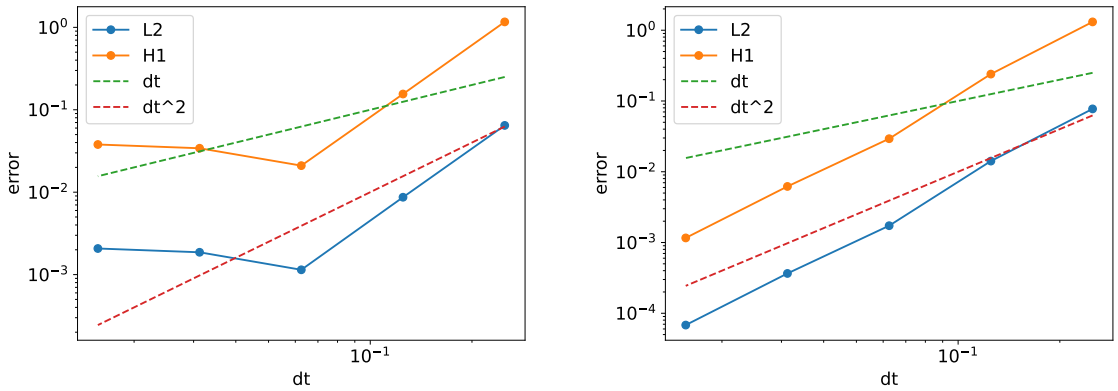
Exercise 2.

Let $\Omega = (0,1)^3$ be the unit cube and $T = 1$. Let us consider the following nonlinear, time-dependent problem:

$$\begin{cases} \dfrac{\partial u}{\partial t} - \boldsymbol{\nabla} \cdot \left(\left(\mu_0 + \mu_1 u^2 \right) \boldsymbol{\nabla} u \right) = f & \text{in } \Omega \times (0,T), \quad (2a) \\ u = g & \text{on } \partial\Omega \times (0,T)\,, \quad (2b) \\ u = u_0 & \text{in } \Omega \times \{0\}, \quad (2c) \end{cases}$$

with $\mu_0 = 0.1$, $\mu_1 = 1$, $g(\mathbf{x}, t) = 0$, $u_0(\mathbf{x}) = 0$ and

$$f(\mathbf{x}, t) = \begin{cases} 2 & \text{if } t < 0.25\,, \\ 0 & \text{if } t \geq 0.25\,. \end{cases}$$



(a) (b)

Figure 2: Error in the L^2 and H^1 norms against Δt, using linear (a) and quadratic (b) polynomials on mesh `mesh/mesh-cube-20.msh` using the Crank-Nicolson method ($\theta = 0.5$).

2.1. Implement in `deal.II` a finite element solver for problem (2), using the implicit Euler method for time discretization and Newton's method for linearization. Then, compute the solution using the mesh `mesh/mesh-cube-20.msh`, with linear finite elements (degree $r = 1$) and $\Delta t = 0.05$.

Solution. We begin by deriving the weak formulation to the problem. Let $V = H_0^1(\Omega)$. At all times $t \in (0, T)$, we look for $u(t) \in V$ such that $u(0) = u_0$ and

$$\int_\Omega \frac{\partial u}{\partial t} v \, d\mathbf{x} + \underbrace{\int_\Omega \left(\mu_0 + \mu_1 u^2\right) \boldsymbol{\nabla} u \cdot \boldsymbol{\nabla} v \, d\mathbf{x}}_{b(u)(v)} = \underbrace{\int_\Omega f \, v \, d\mathbf{x}}_{F(v)} \, .$$

Defining

$$R(u)(v) = \int_\Omega \frac{\partial u}{\partial t} v \, d\mathbf{x} + b(u)(v) - F(v) \, ,$$

the weak formulation reads:

$$\text{for all } t \in (0, T), \text{ find } u(t) \in V \text{ such that } R(u)(v) = 0 \text{ for all } v \in V \text{ and } u(0) = u_0 \, .$$

We now discretize the weak formulation, first in space (using finite elements), then in time (using the implicit Euler method). Let us introduce a mesh over Ω, and let $V_h = V \cap X_h^r(\Omega)$ be the finite element space. The semi-discrete formulation reads:

$$\text{for all } t \in (0, T), \text{ find } u_h(t) \in V_h \text{ such that } R(u_h)(v_h) = 0 \text{ for all } v_h \in V_h \text{ and } u_h(0) = u_{0,h} \, .$$

Then, we introduce the temporal discretization. Let us partition the interval $(0, T]$ into the sub-intervals $(t_n, t_{n+1}]$, with $n = 0, 1, \ldots, N_T - 1$, $t_0 = 0$, $t_{N_T} = T$ and $t_{n+1} - t_n = \Delta t$.

We denote with a superscript n the approximate solution at time t_n, i.e. $u_h^n \approx u_h(t_n)$. The fully discrete formulation of the problem can be expressed as:

$$\underbrace{\int_\Omega \frac{u_h^{n+1} - u_h^n}{\Delta t} v_h \, d\mathbf{x} + b(u_h^{n+1})(v_h) - F^{n+1}(v_h)}_{R^{n+1}(u_h^{n+1})(v_h)} = 0 \qquad \text{for all } v_h \in V_h, n = 0, 1, \ldots, N_T - 1 \tag{3}$$

This is a nonlinear problem, due to b being nonlinear in u_h^{n+1}. Therefore, we employ Newton's method for its solution. To this end, we compute the derivative of the discrete residual $R^{n+1}(u_h^{n+1})(v_h)$:

$$\begin{aligned} a(u_h^{n+1})(\delta_h, v_h) &= \int_\Omega \frac{\delta_h}{\Delta t} v_h \, d\mathbf{x} + \frac{\mathrm{d}b}{\mathrm{d}u}(\delta_h, v_h) \\ &= \int_\Omega \frac{\delta_h}{\Delta t} v_h \, d\mathbf{x} + \int_\Omega (2\mu_1 u_h^{n+1} \delta_h) \boldsymbol{\nabla} u_h^{n+1} \cdot \boldsymbol{\nabla} v d\mathbf{x} + \int_\Omega (\mu_0 + \mu_1 (u_h^{n+1})^2) \boldsymbol{\nabla} \delta_h \cdot \boldsymbol{\nabla} v_h d\mathbf{x} \, . \end{aligned}$$

Therefore, to solve problem (1), we proceed as follows: given the initial solution u_h^0, we iterate for $n = 0, 1, 2, \ldots, N_T - 1$ and compute u_h^{n+1} by solving (3) using Newton's method. At any fixed time n, Newton's method reads: given the initial guess $u_h^{n+1,(0)} = u_h^n$, iterate for $k = 0, 1, 2, \ldots$ and until convergence:

1. assemble and solve the linear problem: $a(u_h^{n+1,(k)})(\delta_h^{(k)}, v_h) = -R(u_h^{n+1,(k)})(v_h)$ for all $v_h \in V_h$;
2. set $u_h^{n+1,(k+1)} = u_h^{n+1,(k)} + \delta_h^{(k)}$.

The problem at step 1 is solved using finite elements by assembling and solving the associated linear system.

In other words, in order to solve problem (2), we need to perform two nested loops. The outer loop iterates through time (i.e. over the index n). The inner loop applies Newton's method (i.e. iterates over the index k). Within the inner loop, we repeatedly assemble J and $\mathbf{r}$ and solve the associated linear system, until a convergence criterion is satisfied.

See the files `src/lab-08-exercise2.cpp`, `src/HeatNonLinear.hpp` and `src/HeatNonLinear.cpp` for the implementation. Figure 3 shows some snapshots of the solution.

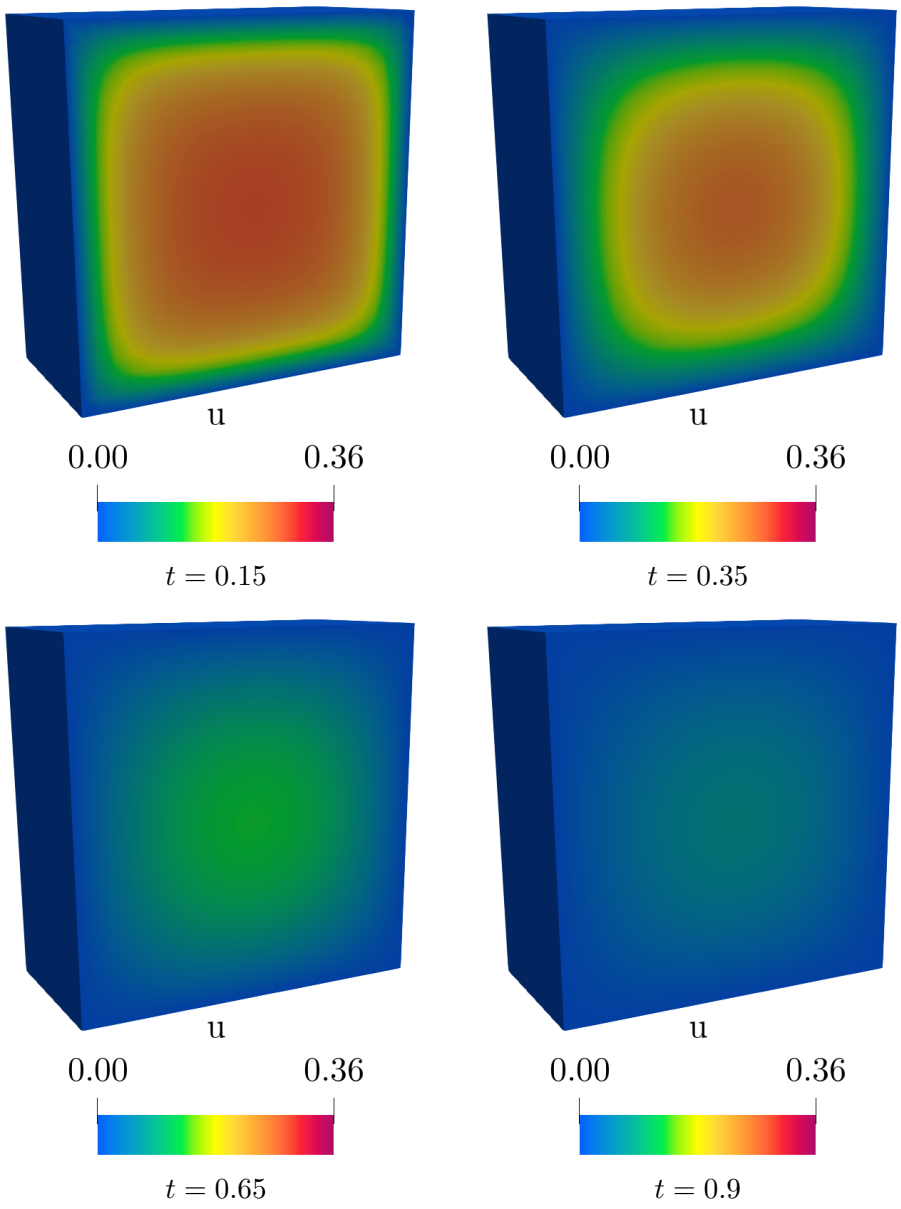


Figure 3: Snapshots of the solution to Exercise 2, clipped along a plane perpendicular to the z axis.