INF5620 — Project 3
Nonlinear diffusion equation

Emilie Fjørner
e.s.fjorner@fys.uio.no

Jonas van den Brink
j.v.d.brink@fys.uio.no

November 29, 2013

Description

In this problem we solve a nonlinear diffusion equation model. We sample the given partial differential equation in time and approximate the time derivative using the Crank-Nicolson method, this leads to an implicit numerical scheme. For each time step we have to solve a non-linear spatial PDE, which is done using the finite element method through FEniCS. The PDE is linearized using the Picard iteration method. A solver is implemented and verified using two test problems. Finally the solver is used to look at the diffusion of a Gaussian function.

The Partial Differential Equation

Formulated as a PDE problem, the nonlinear diffusion model we will be solving has the following form:

$$\rho u_t = \nabla \cdot (\alpha(u)\nabla u) + f(\boldsymbol{x}, t) \text{ on } \Omega, \tag{1}$$

$$\frac{\partial u}{\partial n} = 0 \qquad \text{on } \delta\Omega. \tag{2}$$

We are solving for the scalar field u. The coefficient ρ is a known, real constant and α is a known function of the solution u, making the equation nonlinear. The domain Ω has the boundary $\delta\Omega$ and we see that we have a Neumann boundary condition for the entire boundary.

a) Variational Form

We sample our PDE at the time t_n and use the Crank-Nicolson scheme to approximate the time-derivative. This gives us the following implicit scheme

$$\rho\frac{u^{n+1} - u^n}{\Delta t} = \frac{1}{2}\Big(\nabla \cdot (\alpha(u^{n+1})\nabla u^{n+1}) + f^{n+1} + \nabla \cdot (\alpha(u^n)\nabla u^n) + f^n.\Big)$$

We want to reformulate the PDE as a variational problem. To acheive this we multiply by a test function v, and integrate over the entire spatial domain Ω

$$\begin{aligned}\frac{2\rho}{\Delta t}\Big(\int_\Omega u^{n+1}v \text{ d}x - \int_\Omega u^n v \text{ d}x\Big) = &\int_\Omega \nabla \cdot (\alpha(u^{n+1})\nabla u^{n+1})v \text{ d}x \ + \int_\Omega f^{n+1}v \text{ d}x \\ &+ \int_\Omega \nabla \cdot (\alpha(u^n)\nabla u^n)v \text{ d}x \ + \int_\Omega f^n v \text{ d}x\end{aligned}$$

Using integration by parts on the integrals containing the double derivative of the trial function u, and our Neumann boundary condition this becomes

$$\begin{aligned}\frac{2\rho}{\Delta t}\Big(\int_\Omega u^{n+1}v \text{ d}x - \int_\Omega u^n v \text{ d}x\Big) = &-\int_\Omega \alpha(u^{n+1})\nabla u^{n+1} \cdot \nabla v \text{ d}x \ + \int_\Omega f^{n+1}v \text{ d}x \\ &- \int_\Omega \alpha(u^n)\nabla u^n \cdot \nabla v \text{ d}x \ + \int_\Omega f^n v \text{ d}x\end{aligned}$$

b) and c) Picard iteration

We now have a nonlinear system to solve, and we want to use a Picard iteration method to reformulate this as a linear problem. Renaming the solution to be found, u^{n+1} as simply u, and the solution at the previous time step u_p, we use u_p as our initial guess for the Picard iteration. Our formulation becomes

$$\frac{2\rho}{\Delta t}\Big(\int_\Omega u^{k+1}v\ \mathrm{d}x - \int_\Omega u_p v\ \mathrm{d}x\Big) = \int_\Omega \nabla\cdot(\alpha(u^k)\nabla u^{k+1})v\ \mathrm{d}x\ + \int_\Omega fv\ \mathrm{d}x \\ + \int_\Omega \nabla\cdot(\alpha(u_p)\nabla u_p)v\ \mathrm{d}x\ + \int_\Omega f_p v\ \mathrm{d}x$$

with $k^0 = u_p$.

We restrict this Picard iteration to a single iteration. This is equivalent to simply letting the argument of the α-function be u_p in both of the integrals where it appears.

$$\frac{2\rho}{\Delta t}\Big(\int_\Omega uv\ \mathrm{d}x - \int_\Omega u_p v\ \mathrm{d}x\Big) = \int_\Omega \nabla\cdot(\alpha(u_p)\nabla u)v\ \mathrm{d}x\ + \int_\Omega fv\ \mathrm{d}x \\ + \int_\Omega \nabla\cdot(\alpha(u_p)\nabla u_p)v\ \mathrm{d}x\ + \int_\Omega f_p v\ \mathrm{d}x$$

d) Convergence rate study

As a first verification of our FEniCS implementation we set $\alpha(u) = 1$, $f = 0$, $I(x, y) = \cos(\pi x)$. We solve the equation on the two-dimensional domain $\Omega = [0, 1] \times [0, 1]$ using P1 elements. The PDE then has the known solution

$$u(x, y, t) = e^{-\pi^2 t}\cos(\pi x).$$

To test the convergence rate, we solve for many different values of

$$h = \Delta t^2 = \Delta x^2,$$

and calculate the error at the time $T = 3.0$. The results are shown in table 1

From the results we see that as h is lowered, the error is decreasing quite smoothly and that E/h remains approximately constant.

e) Method of manufactured solutions

We now solve for the exact solution

$$u_e(x, t) = tx^2(\frac{1}{2} - \frac{x}{3}),$$

with $\alpha = 1 + u^2$ on the one-dimensional domain $\Omega = [0, 1]$. Through the use of sympy we compute the corresponding source term that yields the given exact solution. We solve with the resulting source term and find the error of the computed solution by comparing with the known exact solution at the times $T = 0.1$, $T = 0.5$, $T = 1.0$ again we solve for different values of h. The results are shown in table 2. We see that the error decreases with h as expected, but also that it growns significantly with time. For the value $h = 0.01$ we seem to get a very good error, though no clear reason is apparent.

h	E	E/h
1.0e+00	1.563e-02	1.563e-02
5.0e-01	3.580e-05	7.161e-05
1.0e-01	1.816e-04	1.816e-03
5.0e-02	4.147e-05	8.293e-04
1.0e-02	1.951e-06	1.951e-04
5.0e-03	4.624e-07	9.248e-05
1.0e-03	1.862e-08	1.862e-05

Table 1: Measure of error as h is decreased in exercise d).

h	$E(T = 0.1)$	$E(T = 0.5)$	$E(T = 1.0)$
5.0e-01	1.054e-02	1.294e-03	6.815e-03
1.0e-01	1.300e-03	4.936e-04	2.020e-03
5.0e-02	4.247e-04	8.161e-04	3.999e-04
1.0e-02	1.552e-13	3.785e-10	8.154e-09
5.0e-03	4.767e-05	1.202e-04	1.037e-04
1.0e-03	1.950e-05	1.000e-06	4.065e-06

Table 2: Measure of error at different times as h is decreased in exercise e)

f) Sources of numerical error

There are mainly three sources of numerical errors in the FEniCS program. There is an error proportional to Δt^2 as a result of our use of the Crank-Nicolson finite difference approximation in time. There is an error resulting from the finite element method used to solve the spatial PDE problem for each time step, this error will be different for different problems, but for our two test cases it wass proportional to $\Delta x^2 + \Delta y^2$. Finally we have an error due to linearizing our PDE through the use of Picard iterations, this error decreases with an increasing number of Picard iterations k, but as we have limited ourselves to a single iteration per time step the error should contribute significantly. The error due to linearizing the PDE should decrease as the temporal resolution is refined.

h) Diffusion of a Gaussian function

We now use the implemented solver to solve the nonlinear diffusion equation for the initial condition

$$I(x, y) = \exp\left[\frac{1}{2\sigma^2}\left(x^2 + y^2\right)\right], \qquad (x, y) \in \Omega = [0, 1] \times [0, 1],$$

with $\alpha(u) = 1 + \beta u^2$, for some constant β. The resulting field at some time steps are shown in figure 1

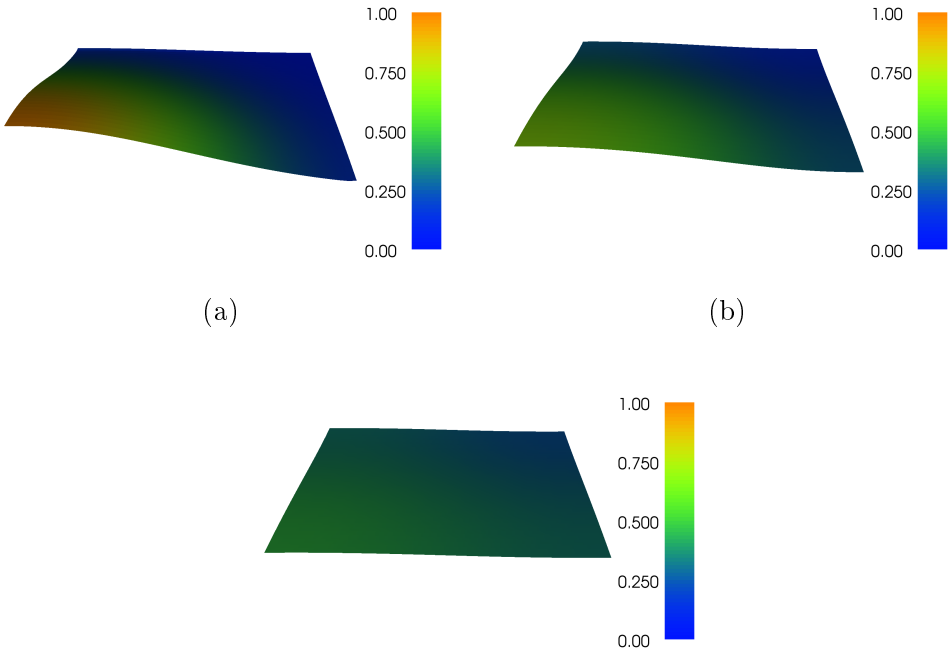

Figure 1: Diffusion of a Gaussian function. (a) At the time $t = 0$, the field $u(x, y)$ is a perfect Gaussian function, though only one quartile is shown. (b) At some later time, the Gaussian function has started to diffuse out. (c) After a relatively long time, the field has become nearly uniform.