INF5620 - Project 2

Steffen Brask

November 3, 2016

a)

We are going to discretize, and solve using FEniCS the equation

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

with initial condition $u(\vec{x}, 0) = I(\vec{x})$ and boundary condition $\frac{\partial u}{\partial n} = 0$. First we discretize in time using a Backward Euler scheme, this gives

$$\rho \frac{u^n - u^{n-1}}{\Delta t} = \nabla \cdot (\alpha(u^n)\nabla u^n) + f(\vec{x}, t) \tag{2}$$

And second we can derive a variational formulation of the spatial part, multiplting with v and partially integrating the $\nabla \cdot (\alpha(u^n)\nabla u^n)$ term we get

$$\rho \frac{u^n - u^{n-1}}{\Delta t} v = \left[\alpha(u^n)\nabla u^n \cdot \nabla v - \alpha(u^n)\frac{\partial u^n}{\partial n}\Big|_{\partial\Omega} \cdot v\right] + f(\vec{x}, t) \cdot v \tag{3}$$

The boundary condition kills the second term inside the square bracket, and rearanging on the form $a(u, v) = L(v)$

$$\rho(u, v) - \Delta t(\alpha(u^n)\nabla u^n, v) = \rho(u^{n-1}, v) + \Delta t(f, v) \tag{4}$$

Where i used the same notation as FEniCS uses for an inner product. For the initial condition we have that

$$\begin{aligned} R_0 =& u(\vec{x}, 0) - I(\vec{x}) \\ we\ require \int R_0 v d\vec{x} = 0 \Rightarrow& \int (u^0 - I) v d\vec{x} = 0, \\ \Rightarrow& (u^0, v) = (I, v) \end{aligned} \tag{5}$$

So we can get u^0 by giving it the same relation to v as I has, i.e interpolation.

b)

There is one more problem to solve before we can program. And that is that the function $\alpha(u)$ uses the unknown u and we need this to solve the equation, so we need u to find u. This can be done by Picard Iterations.

$$\rho(u,v) - \Delta t(\alpha(u^-)\nabla u^n, v) = \rho(u^{n-1}, v) + \Delta t(f,v) \tag{6}$$

Assuming u^{n-1} is known we can solve for u^n and iterate over u^- putting the last u^n in for u^- to find the next. The first iteration we use the best guess, that will be $u^{n-1} = u^-$.

c,d,e,f)

Implementation is done in the program `nonlin_diffus_pde.py`

g)

All of the regular numerical error sources may occur, such as division multiplication errors. In addition we have a global error from the numerical scheme that goes as $O(dt)$. When the function $\alpha(u)$ is not a constant we get errors from the Picard Iteration. And of course there may be some errors in the programming.