NUMERICAL METHODS FOR PARTIAL DIFFERENTIAL EQUATIONS

PROJECT 2

ABINAYA A. SAKTHIVEL

University of Oslo

17. NOVEMBER 2019

1 Mathematical problem

This time we are going to solve nonlinear diffusion equation on the form,

$$\varrho u_t = \nabla \cdot (\alpha(u)\nabla u) + f(\vec{x}, t) \tag{1}$$
$$u(\vec{x}, 0) = I(\vec{x}), \tag{2}$$
$$\frac{\partial u}{\partial n} = 0, \tag{3}$$

in both one, two and three dimension. ϱ is constant and $\alpha(u)$ is a known function of u. Eq. 2 and 3 is the initial condition and boundary condition, respectively.

2 Defining the variational form

First we want to get rid off term that are time dependent using finite difference method. In this case we take use of Backward Euler scheme

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

We then move all the terms to one side and find the residual,

$$R = u^n - u^{n-1} - \frac{\Delta t}{\varrho}\left[\nabla \cdot (\alpha(u^n)\nabla u^n) + f^n(\vec{x}, t)\right]$$

Using the Galerkin method in function space $\boldsymbol{V}$, we integrate over a domain Ω generally on the form [0,L] in 1D.

$$\int_\Omega Rv\, dx = 0 \quad \forall v\, \epsilon\, \boldsymbol{V}$$

$$\int_\Omega u^n v\, dx = \int_\Omega u^{n-1} v\, dx + \frac{\Delta t}{\varrho}\int_\Omega \left[\nabla \cdot (\alpha(u^n)\nabla u^n) + f^n(\vec{x}, t)\right] v\, dx, \quad \forall v\, \epsilon\, \boldsymbol{V}$$

The whole point of finding the variational (weak) form is to get rid of the second order derivative. Thus, we take use of integration by parts

$$\int_\Omega \nabla \cdot (\alpha(u^n)\nabla u^n) v\, dx = -\int_\Omega \alpha(u^n)\nabla u^n \nabla v\, dx + \int_{\partial\Omega} \alpha(u^n)\nabla u^n v\, dx = -\int_\Omega \alpha(u^n)\nabla u^n \nabla v\, dx$$

where $\partial\Omega$ is the boundary of the domain. Due to Neumann boundary condition the last term vanishes and the equation is now on variational form. We rewrite $u^n \to u$ and $u^{n-1} \to u^{(1)}$

$$\int_\Omega uv\, dx = \int_\Omega u^{(1)} v\, dx - \frac{\Delta t}{\varrho}\int_\Omega \alpha(u)\nabla u \nabla v\, dx + \frac{\Delta t}{\varrho}\int_\Omega f(\vec{x}, t) v\, dx \tag{4}$$

For the first step we simply replace u^1 with the initial condition $I(\vec{x})$.

2.1 Picard iteration

In eq. 4 the second term on the right side is nonlinear since α is dependent of u. To solve this equation we first need to make this nonlinear term, linear. This is done by replacing u in α with u^k which is an approximation to the unknown u.

Suppose u^k is the most recently computed approximation to u^n. We then want to find u^{k+1} which is hopefully a better approximation

$$\int_\Omega u^{k+1} v \, dx = \int_\Omega u^{(1),k} v \, dx - \frac{\Delta t}{\varrho} \int_\Omega \alpha(u^k) \nabla u^{k+1} \nabla v \, dx + \frac{\Delta t}{\varrho} \int_\Omega f(\vec{x}, t) v \, dx \tag{5}$$

To get the first iteration started we set $u^{k=0} = u^{(1)}$. After solving the first iteration, u^k is set to be the solution of the previous iteration.

We need to stop this iteration at some point, this is done by requiring some criteria,

$$\left| u - u^k \right| < \epsilon_u$$

where ϵ_u is a chosen tolerance. The other method is to put the latest solved u^k into the residual R and see if it is smaller than a chosen tolerance ϵ_R.

2.2 Variational form in FEniCS

For any timestep n we now have a linear equation to solve (5). We approximate $\alpha(u) \to \alpha(u^{(1)})$ and rearrange the terms in the equation

$$\int_\Omega \left(uv + \frac{\Delta t}{\varrho} \alpha(u^k) \nabla u^{k+1} \nabla v \right) dx = \int_\Omega \left(u^{(1)} v + \frac{\Delta t}{\varrho} f v \right) dx \tag{6}$$

$$\int_\Omega a(u, v) \, dx = \int_\Omega L(v) \, dx$$

where,

$$a(u, v) = uv + \frac{\Delta t}{\varrho} \alpha(u^k) \nabla u^{k+1} \nabla v$$

$$L(v) = u^{(1)} v + \frac{\Delta t}{\varrho} f v$$

3 Verification

3.1 Constant solution

The first verification is to see if a constant solution is reproduced by the implementation. Inserting $u(\vec{x}, t) = C$ into eq. 1 gives a source term, $f(\vec{x}, t) = 0$. ϱ and $\alpha(u)$ can then be chosen arbitrarily since both parameters will disappear anyways. The only parameter that has an effect is the initial condition, $I(\vec{x}) = C$, where C is an arbitrary value. (In my case, C=7)

```
Using P1 elements in 1D gives the absolute error: 1.305622e-13

Using P1 elements in 2D gives the absolute error: 2.140510e-13

Using P1 elements in 3D gives the absolute error: 3.641532e-14
```

The errors are of order 10^{-13} and e^{-14} which indicate on a good numerical implementation.

3.2 Analytical solution

Next, we test if the linear equation works as expected. We set $\alpha = 1$, $f = 0$ and the initial condition to $I(\vec{x}) = cos(\pi x)$.

We have been given a model for error measure

$$E = K_t \Delta t^p + K_x \Delta x^2 + K_y \Delta y^2 = Kh$$

if $h = \Delta t^p = \Delta x^2 = \Delta y^2$. (p=1 for Backward Euler scheme) This gives the relation,

$$N_x = N_y = \frac{L_x}{\Delta x} = \frac{L_y}{\Delta y} = \frac{1}{\sqrt{\Delta t}}$$

```
h=0.1000, E/h=0.1488, N=3

h=0.0500, E/h=0.1267, N=4

h=0.0250, E/h=0.1187, N=6

h=0.0125, E/h=0.1086, N=8

h=0.0063, E/h=0.1075, N=12

h=0.0031, E/h=0.1052, N=17

h=0.0016, E/h=0.1054, N=25

h=0.0008, E/h=0.1041, N=35
```

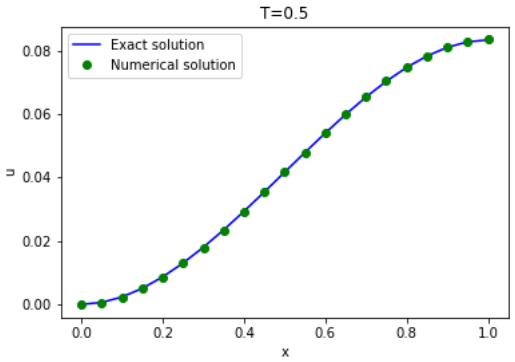
Above we see that the E/h is approximately constant as the mesh in space and time is refined.

3.3 Nonlinear diffusion PDE

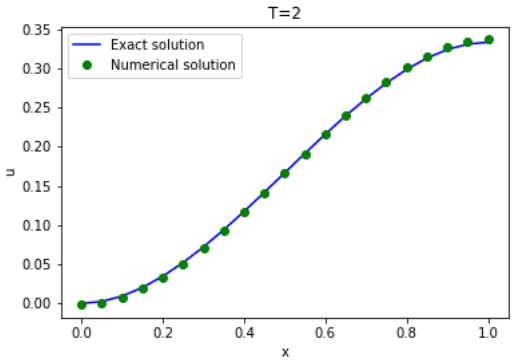
Previous section we tested a linear version without a source term. To check if the implementation of the nonlinear diffusion PDE is correct or not, we use the method of manufactured solutions.

We have been given the source term in the project description, with its corresponding solution u.

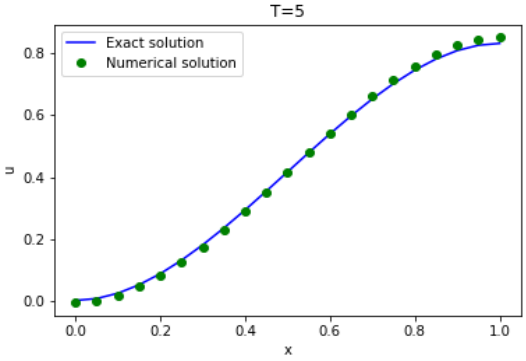
Figures show the comparison between numerical and exact solutions for three different T values.



(a) T=0.5 and dt=0.5



(b) T=2.0 and dt=0.5



Figur (1) T=5.0 and dt=0.5.

4 Numerical errors in FEniCS

1. Error produced by discretization in space and time; $O(\Delta x^2)$, $O(\Delta y^2)$, $O(\Delta z^2)$, $O(\Delta t)$

2. For not so small Δt, the single Picard iteration will contribute to an error.

3. Numerical integration in FEniCS will also lead to errors, but the error in numerical integration formulas is of the same order or smaller than the discretization errors. (Thus, not so big impact)